IDENTIFICATION

PRODUCT CODE: AH-E???A-DD

DIAGNOSTIC CODE: DFPTA

PRODUCT NAME: DFPTADO Port Basic Device Diagnostic

VERSION: 0.4

DATE RELEASED: May 15, 1985

MAINTAINED BY: Peripherals/CPS Engineering

AUTHOR: Richard Stockdale

UPDATE AUTHOR: Gregory A. Scott

COPYRIGHT (C) 1983, 1985

DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DIGITAL EQUIPMENT CORPORATION.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE IN EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL EQUIPMENT CORPORATION.

Table of Contents

Page

1.0 PROGRAM ABSTRACT 1

2.0 REQUIREMENTS 2
2.1 Hardware Requirements 2
2.1.1 KL10 Microcode Requirements 2
2.1.2 Memory Requirements 2
2.1.3 Load Device 2
.2 2Software Requirements 2
2.3 Diagnostic Prerequisites 3

3.0 PROGRAM LOADING PROCEDURE 4

4.0 PROGRAM STARTING PROCEDURE 5
4.1 Starting the Diagnostic 5
4.2 Initial Program Switches 5
4.3 Startup Procedure 5
4.3.1 Port Configuration 7
4.4 Optional Restart Address 7

5.0 PROGRAM STOPPING PROCEDURE 8

6.0 PROGRAM OPERATING PROCEDURE 9
6.1 Overview 9
6.2 Configuration 9
6.2.1 Configuration Report 9
6.2.2 Configuration Modification 9
6.2.3 Reconfiguration Capability 10
6.3 Choosing Run Time Options 10
6.4 Modes of Diagnostic Operation 10
6.5 Scope Looping 10
6.6 General User Interface Philosophy 10
6.6.1 User Interface 11
6.7 Terminal Input 11
6.7.1 General 11
6.7.2 Control-C Capability (Ctrl-C) 11
6.7.3 Control-O Capability (Ctrl-O) 11
6.7.4 Control-Q Capability (Ctrl-Q) 12
6.7.5 Control-S Capability (Ctrl-S) 12
6.7.6 Control-T Capability (Ctrl-T) 12
6.7.7 Control-U Capability (Ctrl-U) 12
6.7.8 Control-Z Capability (Ctrl-Z) 12
6.7.9 Question Mark (?) 12
6.7.10 Altmode (ESC) 13
6.8 Terminal Output 13

7.0 PROGRAM OPTIONS (SWITCHES) 14
7.1 Left Hand Switches 14
7.1.1 Left Hand Switches (Brief Description) 15
7.1.2 Left Hand Switches (Detailed Description) 16
7.2 Right Hand Switches 19
7.2.1 Right Hand Switches (Brief Description) 19
7.2.2 Right Hand Switches (Detailed Description) 20

8.0 ERRORS 24
8.1 User Input Errors at the Terminal 24
8.2 Error Reporting of Hardware Detected Errors . . 24
8.2.1 Reporting to SPEAR 24
8.2.2 Error Printouts 25
8.3 Unexpected Errors 26

9.0 PROGRAM DESCRIPTION 27
9.1 Program Overview 27
9.1.1 Program Purpose 27
9.1.2 Testing Strategy 27
9.2 Run-Time Dynamics 27
9.2.1 Memory Allocation Requirements 27
9.2.2 Operating System Interactions 28
9.2.2.1 TOPS-10 28
9.2.2.2 TOPS-20 28
9.2.3 User Mode Restrictions 28
9.2.3.1 User Mode Privileges 29
9.2.3.2 User Mode Priority Interrupts 29
9.3 Exec Mode Capability 29
9.4 User Mode Capability 29
9.5 Fault Detection 29
9.5.1 Error Resolution 29
9.5.2 Fault Detection And Isolation 29
9.5.3 Fault Detection 30
9.6 Network Descriptions 30
9.6.1 EBUS Module networks 30
9.6.1.1 E1 - EBUS Control Logic 30
9.6.1.2 E2 - CC EBUS RQST 30
9.6.1.3 E3 - CC GRNT CSR 30
9.6.1.4 E4 - Interrupt Logic 31
9.6.1.5 E5 - Clock Logic 31
9.6.1.6 E6 - Clear Logic 31
9.6.1.7 E7 - Magic # Field Decode 31
9.6.1.8 E8 - CC CSR CHNG 31
9.6.1.9 E9 - KMUX 31
9.6.1.10 E10 - EMUX 31
9.6.1.11 E11 - RMUX/CSR 25-31 32
9.6.1.12 E12 - CSR 11-13/19-22/33-35 32
9.6.1.13 E13 - CSR 04 Rqst Exam/Dep 32
9.6.1.14 E14 - CSR 05 Rqst Interrupt/CC INTR ACTIVE . 32
9.6.1.15 E15 - CSR 06/07 CRAM PE/MBUS Error 32
9.6.1.16 E16 - CSR 24 EBUS PE 32
9.6.1.17 E17 - CSR 32 MPROC Run 32
9.6.1.18 E18 - KMUX Parity Generation 33
9.6.1.19 E19 - EBUS Parity Checking 33
9.6.1.20 E20 - EBUF 33
9.6.1.21 E21 - EBUS Data Xcvrs 33
9.6.1.22 E22 - EBUS Control Xcvrs 33
9.6.1.23 E23 - 2901s/Carry Logic 33
9.6.1.24 E24 - Constant MUX 33
9.6.2 MPROC Module networks 33
9.6.2.1 M1 - MPROC Run Clocks 34
9.6.2.2 M2 - Local Memory Control 34
9.6.2.3 M3 - MBUS Error Detect Logic 34
9.6.2.4 M4 - CRAM Read/Write Control 34

9.6.2.5 M5 - Jump ADDR MUX 34
9.6.2.6 M6 - CC MUX 34
9.6.2.7 M7 - 2910 MicroSequencer 34
9.6.2.8 M8 - CRAM ADDR MUX 34
9.6.2.9 M9 - LAR 35
9.6.2.10 M10 - CRAM 0000-7777 35
9.6.2.11 M11 - Cond/Skip Decoder 35
9.6.2.12 M12 - CRAM Control Register 35
9.6.2.13 M13 - Local Storage RAM 35
9.6.2.14 M14 - Local Storage Addr Register 35
9.6.2.15 M15 - RAM Mode MUX 35
9.6.2.16 M16 - RAR 35
9.6.2.17 M17 - Right CRAM Load Buffer 36
9.6.2.18 M18 - Left CRAM Load Buffer 36
9.6.2.19 M19 - MW Out MUX 36
9.6.3 CBUS Module networks 36
9.6.3.1 C1 - CBUS Control Logic 36
9.6.3.2 C2 - Clock Logic 36
9.6.3.3 C3 - Reset Logic 36
9.6.3.4 C4 - Clear Logic 36
9.6.3.5 C5 - "CC CBUS AVAIL" 36
9.6.3.6 C6 - "CC CB LST WD"/ "CC CHAN ERR" 37
9.6.3.7 C7 - "CLR CC CODE" 37
9.6.3.8 C8 - Magic # Field Decode 37
9.6.3.9 C9 - "CC CBUS PAR ERR" 37
9.6.3.10 C10 - "CC PLI PAR ERR" 37
9.6.3.11 C11 - Parity Predictor 37
9.6.3.12 C12 - Formatter Control 37
9.6.3.13 C13 - PLI Control Logic 37
9.6.3.14 C14 - CBUS Input Buffer 38
9.6.3.15 C15 - CBUS Output Buffer 38
9.6.3.16 C16 - CBUS Out Par Gen 38
9.6.3.17 C17 - CBUS In Par Checker 38
9.6.3.18 C18 - DMUX 38
9.6.3.19 C19 - SMUX 38
9.6.3.20 C20 - Mvr/Fmtr 38
9.6.3.21 C21 - PLI Logic 38
9.6.3.22 C22 - PMUX 38
9.6.3.23 C23 - CMUX 39
9.6.3.24 C24 - CBUF 39
9.7 Performance During Hardware Failures 39
9.7.1 CPU/Memory Failures 39
9.7.2 Power Fail/Restart 39
9.7.3 Port Failure 39
9.7.4 Unexpected Traps 39
9.8 Program Applications 39
9.8.1 Users and Uses 40
9.8.2 Engineering Usage 40
9.8.3 Manufacturing Usage 40
9.8.4 Customer Services Usage 40
9.8.5 Customer Usage 40
9.9 Test Design 40
9.9.1 Type of Tests 41
9.9.2 Test Flow 42
9.9.2.1 Starting the Test 42
9.9.2.2 Stopping the Test 42

9.9.2.3 Obtaining Run Time Status (Control-T) . . . 43
9.9.2.4 Obtaining Run Time Status (S or Control-S) . 43

10.0 TEST DESCRIPTIONS 44
10.1 Test Description Assumptions 44
10.2 Test Description Overview 44
10.3 Program Option Selections 45
10.3.1 Basic Description 45
10.3.2 Commands Available 45
10.3.3 Detailed Command Description 46
10.3.3.1 Hardware Tests 46
10.3.3.2 General Commands 48
10.3.3.3 Test Control Commands 49
10.3.3.4 Switches 51
10.4 Hardware Tests 53
10.4.1 Hardware Tests - EBUS Module 53
10.4.1.1 EBUS Interface Tests 53
10.4.1.2 CSR Register Tests 53
10.4.1.3 EBUF Tests (MBus Loopback) 53
10.4.1.4 CSR Bit 18 "Clear Port" Tests 53
10.4.1.5 "Diag Test Ebuf"/Initialized 53
10.4.1.6 Parity Generators/Checkers Tests 54
10.4.1.7 CSR Read Only Bits 54
10.4.1.8 CSR 01 "DIAG RQST CSR" 54
10.4.1.9 CSR 02 "DIAG CSR CHNG" 54
10.4.1.10 Contant MUX 54
10.4.1.11 CSR 05 "Rqst Int" Bit 54
10.4.1.12 CSR 06 "CRAM PE" Bit 54
10.4.1.13 CSR 07 "MBUS Err" Bit 55
10.4.1.14 CSR Bits 11,12,13 55
10.4.1.15 MPROC Run Bit 55
10.4.1.16 CSR 22 "Diag Single Cyc" 55
10.4.1.17 PIA Level Selection 55
10.4.1.18 IOP Function Words 55
10.4.1.19 CSR Bits 25-31 55
10.4.1.20 2901 Isolation Tests 56
10.4.1.21 2901 Boolean Tests (Direct Input/0) 56
10.4.1.22 2901 Register File Tests 56
10.4.1.23 2901 ALU Function Tests 56
10.4.1.24 Register Interference Tests 56
10.4.1.25 Q-Register Shifter Tests 57
10.4.1.26 RAM Register Shifter Tests 57
10.4.2 Hardware Tests - MPROC Module 57
10.4.2.1 CRAM Tests 57
10.4.2.2 RAR/LAR Tests 57
10.4.2.3 Cram Ctrl Register Tests 57
10.4.2.4 MBUS Error Tests 57
10.4.2.5 Condition Code Tests 58
10.4.2.6 Local Storage Tests 58
10.4.2.7 Miscellaneous MPROC Tests 58
10.4.2.8 2910 Basic Instruction Tests 58
10.4.3 Hardware Tests - CBUS Module 59
10.4.3.1 Basic CBUS Tests 59
10.4.3.2 CBUS Data Transfer Tests 59
10.4.3.3 Mover/Formatter Tests 60
10.4.3.4 Miscellaneous CBUS Tests 60

10.4.3.5 Parity Predictor Tests 60
10.5 Debug Test 62
10.5.1 Basic Description 62
10.5.2 Commands Available 62
10.5.3 Detailed Command Description 65
10.5.3.1 General Commands 65
10.5.3.2 Device Selection Commands 68
10.5.3.3 Starting, Stopping, Initializing the Port . 69
10.5.3.4 Handling Port Registers 71
10.5.3.5 Handling Port CRAM Locations 73
10.5.3.6 2901 Commands 75
10.5.3.7 Local Storage Commands 76
10.5.3.8 Special Scope Functions 77
10.5.3.9 Mark Bit Commands 78
10.5.3.10 Breakpoint Commands 79
10.5.3.11 Microcode Commands 80
10.5.3.12 Miscellaneous Commands 82
10.5.3.13 Data Transfer Commands 85
10.5.3.14 Switches 86

11.0 PROGRAM RUN TIME 88

1.0 PROGRAM ABSTRACT

DFPTA is the basic device diagnostic for the CI20/NIA20 controller on the KL10. It is intended to detect and isolate hard or stuck at faults in the device. It is used by Manufacturing to repair CI20/NIA20 modules. It is used by Field Service to verify the operation of a port or to isolate a fault to a replaceable module.

DFPTA tests one or two CI20/NIA20 controllers. Each controller consists of three port modules residing in RH20 slot #5 or #7.

DFPTA isolates faults to a network of failing chips. Each network is a set of chips, generally functionally related. Typically, several networks are printed, with the first network being the most probable.

DFPTA consists of two major program sections:

- o Test section - Contains all of the hardware tests. This section is used to debug a module or verify a port.
- o Debug section - Allows detailed manipulation or inspection of the port.

2.0 REQUIREMENTS

2.1 Hardware Requirements

Hardware requirements consist of:

- A KL10, Model B, with at least 93K of memory.
- An CI20/NIA20 Port in RH slot 5 or RH slot 7 or both.
- A load device, such as a KLAD pack.

2.1.1 KL10 Microcode Requirements

The CI20/NIA20 requires special functionality of the KL10 microcode. Therefore, Version 312 or above must be loaded both in user and exec mode.

2.1.2 Memory Requirements

The diagnostic requires 93K of memory of any type. Memory may be configured in any manner, i.e., internal, external, interleaved in any rational manner. The diagnostic's performance is not affected by memory and cache options other than in overall diagnostic run time.

2.1.3 Load Device

The diagnostic loads from any standard load device. Program loading is a function of the standard diagnostic loaders and monitors, i.e., DIAMON, D20MON, MAGMON, KLDCP, etc.

2.2 Software Requirements

Software requirements consist of:

- In exec mode the diagnostic may be loaded by KLDCP, DIAMON, D20MON, or MAGMON. The Subroutine Package must be loaded prior to running the diagnostic.
- In user mode under TOPS-10, release 7.03 or greater is required. DIAMON must be used to load and execute the diagnostic.
- In user mode under TOPS-20, release 6.0 or greater is required. D20MON must be used to load and execute the diagnostic.

2.3 Diagnostic Prerequisites

The diagnostic should only be run after it has been verified that the system has no solid hardware problems of a nature that would render the diagnostic unable to run. This should be determined by first running the following diagnostic programs:

- o CPU and Memory Diagnostics
- o E-Bus Diagnostics
- o C-Bus Diagnostics
- o MAINDEC-10-DFRHB - RH20 Fault Isolator Diagnostic.

3.0 PROGRAM LOADING PROCEDURE

The diagnostic must be loaded via a diagnostic loader program. Refer to loader documentation for instructions on loading and operating the loader program. The diagnostic may be run as a single program or may be chained with other programs using the loader chain feature.

Additionally, in exec mode under KLDCP, the diagnostic may be loaded without using a diagnostic loader program by telling KLDCP to load the diagnostic. This has the advantage of being simpler, but has the disadvantage of inhibiting file handling operations which then prevents use of the TAKE command, and loading of operational microcode by the diagnostic. Loading of operational microcode is required for many of the tests.

4.0 PROGRAM STARTING PROCEDURE

4.1 Starting the Diagnostic

The diagnostic is started in the same manner as most other diagnostics. The diagnostic loader program is run, then the name of the diagnostic, DFPTA, is specified. The loader then loads and starts the diagnostic.

If the diagnostic is already loaded, it may be restarted by a STD or STM in exec mode, or a START or CONTINUE in user mode.

4.2 Initial Program Switches

In exec mode, right hand switches may be selected using the SW command as documented in the KL10 Maintanance Guide under KLDCP. Left hand switches may be selected on the PDP-11 switch panel. If the switches are 0, the diagnostic runs 5 passes of all hardware tests then exits. To allow the user to select which test(s) to run, the left hand switch "OPRSEL" should be set.

In user mode, switches will be queried when the diagnostic starts up. At this point, OPRSEL switch can be set, and any other switches.

In both exec and user mode, switch settings may be altered after the diagnostic has been started.

4.3 Startup Procedure

When the diagnostic is started, the diagnostic name and version number will be printed, followed by a memory map of the system. The diagnostic then looks at RH slots 5 and 7, print what ports are available, and asks the user what ports to test.

If OPRSEL switch is not set, the diagnostic will test every port that exists and is selectable. If this switch is set, the diagnostic will ask what ports to test.

For example (OPRSEL switch not set):

```
D20MON CMD - DFPTA

DFPTA Port Basic Diagnostic
VERSION 0.4, SV=0.15, TOPS-20, KL10, CPU#=1025

SWITCHES = 000000 000000

MEMORY MAP =
FROM     TO          SIZE/K
00000000 00271777      93
```

```
Port Configuration:
  Port  Exists  Selectable  Selected
   NI    Yes       Yes         Yes
   CI    Yes       Yes         Yes

[CI port selected]
[NI port selected]
NI End Pass 1. at 2m 45s
CI End Pass 1. at 5m 29s
NI End Pass 2. at 8m 14s
CI End Pass 2. at 10m 58s
NI End Pass 3. at 13m 43s
CI End Pass 3. at 16m 27s
NI End Pass 4. at 19m 11s
CI End Pass 4. at 21m 56s
NI End Pass 5. at 24m 39s
CI End Pass 5. at 27m 24s

D20MON CMD -
```

For example (OPRSEL switch set):

```
D20MON CMD - DFPTA

DFPTA Port Basic Diagnostic
VERSION 0.4, SV=0.15, TOPS-10, KL10, CPU#=1030

SWITCHES = 000010 000000

MEMORY MAP =
FROM      TO          SIZE/K
00000000 00271777      93

Port Configuration:
  Port  Ex[illegible]  Selectable  Selected
   NI    Y[illegible]       Yes         Yes
   CI    Ye[illegible]      Yes         Yes

Test NI port? (Y [illegible] - N

Test CI port? (Y or N)

[CI port selected]

Type ? for help

What test -
```

The diagnostic asks what device to test, and then asks what test to run. Any test may be selected, or ? or HELP typed to find out what tests are available.

At the end of a test the diagnostic continues at the question "What test -". The next test is selected and run and so on.

4.3.1 Port Configuration

In the "Port Configuration" printout above, "Port Exists" is determined by a CONI. If the "port present" bit is set, then "Port Exists" will be a "Yes". If the port is broken, this bit may not be seen, thereby causing "Port Exists" to be "No". If this happens, the port should be selected for testing anyway.

If "Port Selectable" is "Yes", the port may be selected for testing. Both NI and CI ports are always selectable in exec mode. In user mode a port is selectable only if the diagnostic can take control of the port via a DIAG JSYS.

"Port Selected" is initially "Yes" if the port is selectable. Printing out the configuration at a later time will show which ports have actually been selected.

4.4 Optional Restart Address

Under TOPS-10 or TOPS-20 the diagnostic can be restarted via the "REENTER" command. The program switches are queried, then either testing is begun immediately (if OPRSEL switch is not set) or "What Test - " is asked (if OPRSEL switch is set).

Startup is identical to what occurs if the "START" command is given, except for the DIAMON asking "DIAMON CMD - ".

5.0 PROGRAM STOPPING PROCEDURE

Several methods may be used to stop the diagnostic:

- o Altmode.
- o Control-C.
- o ABORT switch.
- o RSTART switch.

A test can be aborted at any time by typing an altmode on the terminal. When an altmode is typed, the diagnostic is restarted at the test selection routine. At this point, if OPRSEL switch is set, the diagnostic asks what test to run next. If OPRSEL switch is not set the diagnostic runs all tests over again.

Control-C may be typed at any time to stop the diagnostic. To restart, a ''START'', ''CONTINUE'', or ''REENTER'' may be typed in user mode. In exec mode a ''STD'', ''ST'', ''STM'', or ''RN'' may be used.

In exec mode, the ABORT switch may be set, so the diagnostic halts at end of pass.

In exec mode, the RSTART switch may be set. This causes the diagnostic to be restarted, and if OPRSEL switch is set, the diagnostic will have effectively stopped and will be waiting for selection of what test to run.

6.0 PROGRAM OPERATING PROCEDURE

6.1 Overview

The diagnostic is designed to perform repair level testing of the CI20/NIA20 controller.

Essentially, the diagnostic consists of a series of tests which test each of the modules. Some of these are in the form of KL10 based tests using CONO, CONI, DATAO, and DATAIs. The remainder are microcode routines which are loaded into the port CRAM and executed.

6.2 Configuration

When the diagnostic starts up it assumes that a port may be located in RH20 slot #5 or #7. It will select both the possible NI port and the CI port for testing. If user mode, the diagnostic will ask the system for each port. If it does not succeed, it will mark that port as unselectable and will deselect it.

Then the diagnostic prints a port configuration report and asks the user what ports to test. If the port to test has already been selected via switches NIPORT and CIPORT, the diagnostic will not ask.

6.2.1 Configuration Report

The port configuration looks like so:

```
! Port Configuration:
!   Port  Exists  Selectable  Selected
!    NI    No       Yes         No
!    CI    Yes      Yes         Yes
```

This report is printed at diagnostic startup. It can also be obtained at any other time with the command "CONFNT" (Print Configuration), if the switch "OPRSEL" is set.

6.2.2 Configuration Modification

If switch "OPRSEL" is set the selected configuration can be modified using the "SELECT" command to reselect what port to test. Only those ports which are selectable can be selected or deselected.

See Section 10.5.3.2 "Device Selection Commands" for more details concerning this command.

6.2.3 Reconfiguration Capability

If switch "OPRSEL" is set, configuration can be redone using the "CONFIG" command to determine the configuration anew, print out the configuration and ask what port to test.

See Section 10.5.3.2 "Device Selection Commands" for more details concerning this command.

6.3 Choosing Run Time Options

Before starting a test, the diagnostic may ask how many times to execute test selected. If the "OPRSEL" switch is not set this question is not asked, and 5 passes are run on each available port. No other run time options are available, aside from program switch options which are fully described in the Section 7.0 "Program Options (Switches)".

6.4 Modes of Diagnostic Operation

The diagnostic runs in either user mode or exec mode. The operation of the diagnostic is almost identical in either case with the exception of the handling of console switches, speed of operation, and choice of tests to run. The diagnostic runs faster in exec mode for obvious reasons. Also, the PI system is not used in user mode - this means that certain interrupt logic tests do not run in user mode.

In addition, the diagnostic runs with or without user interaction. This is determined by the console switch "OPRSEL". More details may be found in Section 7.0 "Program Options (Switches)".

6.5 Scope Looping

Scope looping is supported by the diagnostic. Several of the program switches described in Section 7.0 "Program Options (Switches)" may be used to cause looping on error and affect error printout when looping.

6.6 General User Interface Philosophy

Decimal numbers are always followed by a point. All other numbers are octal.

All input is preceded with a prompt. All input is checked for legality and incorrect responses are followed by a reprompt and/or an additional error message. All numerical input can be specified as an octal number. Some numerical input may be alternatively given as a decimal number followed by a decimal point. Where the the diagnostic allows multiple input options, a help message is available to explain the options. The diagnostic requests three types of input - numerical, yes or no, and sixbit.

Additionally, the user may obtain a short description of the input expected at any point by typing a question mark followed by a carriage return. The text is printed and the question re-asked.

6.6.1 User Interface

The user interface to the diagnostic is dialogue through the terminal. In exec mode the console switches on the KL are used directly in addition to software switches. In user mode the software switches are used as standard procedure. All user/diagnostic dialogue is done through the diagnostic subroutine package.

6.7 Terminal Input

6.7.1 General

Terminal input is handled by the Subroutine Package. It does not have COMND JSYS style input. However, some control characters are supported. And the diagnostic supports ? and help commands where appropriate to facilitate input.

6.7.2 Control-C Capability (Ctrl-C)

Control-C is used to halt diagnostic execution. In exec mode, control is returned to KLDCP at the ">." prompt. In user mode, the user is returned to monitor level.

When ready to continue, the user may restart or continue the diagnostic in any manner listed previously in Section 5.0, "Program Stopping Procedure". This would be by saying "CONTINUE" in user mode or "RN" in exec mode.

6.7.3 Control-O Capability (Ctrl-O)

Control-O inhibits terminal output. Terminal output resumes when the diagnostic prints some forced output or when the user types another Control-O.

6.7.4 Control-Q Capability (Ctrl-Q)

In exec mode Control-Q is treated as an illegal character by the diagnostic and typing it in response to a question results in an error message.

In user mode Control-Q allows terminal output which may have been inhibited by a prior Control-S. This character may be typed at any time with no other effect than resuming terminal output.

6.7.5 Control-S Capability (Ctrl-S)

In exec mode, Control-S is treated as an illegal character by the diagnostic and typing it in response to a question results in an error message. It may also be typed followed by a carriage return in order to obtain run-time status of the diagnostic.

In user mode, it is used as a terminal output control character along with Control-Q, and has no other function.

6.7.6 Control-T Capability (Ctrl-T)

Control-T is treated as an illegal character by the diagnostic and typing it in response to a question results in an error message in exec mode. In user mode, Control-T gives the current program status and does not interfere with program input.

6.7.7 Control-U Capability (Ctrl-U)

Control-U is used to delete whatever has been typed so far in response to a question. After typing Control-U, the desired input can be entered.

6.7.8 Control-Z Capability (Ctrl-Z)

Control-Z is used to exit from DDT, to either KLDCP in exec mode or to monitor level in user mode.

6.7.9 Question Mark (?)

Question mark may be typed in response to any question. The diagnostic prints a short help message and reprompts the user. If the diagnostic is asking for a test name or a debug command, it prints a list of all valid commands that may be typed.

If a question mark is typed after a legal command, the expected format of the command is given, showing possible arguments.

6.7.10 Altmode (ESC)

Altmode is normally used to abort a test or a sequence of test. If it is given in response to a question, the diagnostic aborts the task which is under way. For example, in DEBUG test, if the user is altering a CRAM location and types altmode in the middle of that, the diagnostic aborts the operation and returns to "DEBUG>" prompt without writing the location.

6.8 Terminal Output

The terminal output generated by the diagnostic consists of:

- Program initialization information
- Program questions/messages
- Test error messages

Much of this output can be affected by program switches, specifically, NOPNT, PNTLPT, TXTINH, and INHMSG.

7.0 PROGRAM OPTIONS (SWITCHES)

Program options are controlled by switches. Left hand switches are predefined but not all are used. Most of the right hand switches are defined specific to the diagnostic.

7.1 Left Hand Switches

In exec mode, left hand switches are modified either via the front end console switches or by using the KLDCP command ESW. See KLDCP documentation for details on the use of the ESW command. Typically, these switches are modified by using the console switches.

In user mode, left hand switches are queried when the diagnostic starts. After that, if the OPRSEL switch is set, switches may be changed using the SWITCH command. The diagnostic will ask for the switches in the same manner as occurred at start up. Also the name of the switch can be typed as a command. This turns the switch on or off depending on its state.

For example:

```
What test - TXTINH

[TXTINH On]

What test - TXTINH

[TXTINH On]

What test - SWPRIN

Switches:  000210 000000   TXTINH OPRSEL

What test - TXTINH

[TXTINH Off]

What test - SWITCH

Print the selectable pgm switches? (Y or N) - N

Switches = 000010 000000

10  LH Switches <# or ?> - 210

0  RH Switches <# or ?> - 0

SWITCHES = 020210 000000

What test - SWPRIN

Switches:  000210 000000   TXTINH OPRSEL
```

7.1.1 Left Hand Switches (Brief Description)

Switch	Symbol	State	Function Description
0 (400000)	ABORT	0	No function.
		1	Abort at end of pass.
1 (200000)	RSTART	0	No function.
		1	Restart diagnostic.
2 (100000)	TOTALS	0	No function.
		1	No function.
3 (040000)	NOPNT	0	Normal typeout.
		1	Inhibit all printing except forced.
4 (020000)	PNTLPT	0	Normal output to TTY.
		1	Print on LPT (user, logical dev).
5 (010000)	DING	0	No function.
		1	Ring bell on error.
6 (004000)	LOOPER	0	Normal operation.
		1	Loop on error.
7 (002000)	ERSTOP	0	Normal operation.
		1	Halt on error.
8 (001000)	PALERS	0	Print only first error in loop.
		1	Print all errors.
9 (000400)	RELIAB	0	No function.
		1	Reliability mode switch.
10 (000200)	TXTINH	0	Print full error messages.
		1	Short error messages.
11 (000100)	INHPAG	0	KL - allow 4096K addressing.
		1	KL - inhibit paging.
12 (000040)	MODDVC	0	No function.
		1	Modify device codes.
13 (000020)	INHCSH	0	Use cache.
		1	Inhibit using cache on pgm start.
14 (000010)	OPRSEL	0	Run default operations.
		1	Operator test selections.
15 (000004)	CHAIN	0/1	Used by ''DIAMON'', etc. to control

			chain operations.
16 (000002)	Unused	0 1	KA10 60 Hertz power KA10 50 Hertz power
17 (000001)	Unused		Reserved

7.1.2 Left Hand Switches (Detailed Description)

The following is a detailed description of each console data switch function:

Switch 0 (ABORT)

When this switch is set, the diagnostic returns control to the diagnostic loader at end of pass.

Switch 1 (RSTART)

When this switch is set while a test is being executed, the diagnostic aborts testing after completion of the current test, and restarts the diagnostic.

Switch 2 (TOTALS)

Setting this switch has no effect on diagnostic operation.

Switch 3 (NOPNT)

This switch inhibits all printout except forced. Forced printout is used with questions requiring user response. This switch is useful when it is desired to force a test to run without printing error information. To suppress the remainder of a current error report, type a Control-O on the terminal. This suppresses the remaining error information for the current error and allows printing to start again for the next error.

Switch 4 (PNTLPT)

When this switch is zero, all printouts from the diagnostic are printed on the terminal. If set only forced printouts are printed on the terminal. All data, including forced printouts, is printed on the line printer.

Switch 5 (DING)

This switch causes the terminal bell to be rung when an error occurs.

Switch 6 (LOOPER)

This switch causes error looping within a test. Looping occurs in the first failing test. The diagnostic continues to loop on the test or a portion thereof even if it stops failing, until the LOOPER switch is reset or an altmode is typed.

Switch 7 (ERSTOP)

This switch causes the diagnostic to terminate testing when an error occurs. If the OPRSEL switch is set the diagnostic will continue at the 'What Test -' prompt. If the OPRSEL switch is not set the diagnostic will exit and return control to the diagnostic loader.

Switch 8 (PALERS)

Normally an error message is printed only the first time an error occurs if that error is being repeated continuously. To print all occurrences of every error, set this switch.

Switch 9 (RELIAB)

If this switch is set, each test is run 4 times before control is passed to the next test selected. If only one test has been selected, the test would be run 4 times the repeat count given for the test.

Switch 10 (TXTINH)

If this switch is not set, complete data is printed with each error message. If this switch is set, less data is printed in a shortened form.

Switch 11 (INHPAG)

This switch inhibits paging if set and allows it if not set.

Switch 12 (MODDVC)

If this switch is set, DIAMON asks for any device code modifications. This is not used by the diagnostic itself.

Switch 13 (INHCSH)

This switch, if set, inhibits the use of cache during the time the diagnostic is run. This switch is inspected only in exec mode and only at diagnostic start up. Changes of the switch setting do not affect the cache after the diagnostic has started up.

Switch 14 (OPRSEL)

If set, this switch enables increased user interaction. The diagnostic asks explicitly what test or function to do before doing anything. Test selection is made in response to the question:

```
What Test? -
```

At least enough letters to recognize the test selection must be typed although up to six letters are accepted. "?" may be typed to get a list of the test names or commands. "HELP" may be typed to get a short description of each test or command.

If this switch is not set, the diagnostic runs 5 passes of all hardware tests on each port, prints fault isolation data (if the inhibit fault isolation switch is not set), then exits.

Switch 15 (CHAIN)

This switch is used by DIAMON/D20MON in chaining operations and is not used by the diagnostic in normal operation.

Switch 16 (50CYC) - Unused

This switch is used on KA10s to select 50 Hertz power. It is not used by this diagnostic.

Switch 17 - Unused

This switch is reserved for use by the diagnostic monitor or subroutine package and is not used by the diagnostic.

7.2 Right Hand Switches

7.2.1 Right Hand Switches (Brief Description)

Switch	Symbol	State	Function Description
18 (400000)	TRACE	0	Normal operation.
		1	Trace operation of tests.
19 (200000)	INHFLT	0	Normal operation.
		1	Inhibit fault isolation.
20 (100000)	INHMSG	0	Normal operation.
		1	Inhibit error message printout.
21 (040000)	LOOPGM	0	Normal operation.
		1	Loop on program.
22 (020000)	LOOPTS	0	Normal operation.
		1	Loop on selected test.
23 (010000)	RUNALL	0	Normal operation.
		1	Run all segments of each test.
24 (004000)	DSPEAR	0	Normal operation.
		1	Disable SPEAR error reporting.
25 (002000)	UDEBUG	0	Normal operation.
		1	Diagnostic debug mode.
26 (001000)	MDEBUG	0	Normal operation.
		1	Error message debug mode.
27 (000400)	SDEBUG	0	Normal operation.
		1	Test execute debug mode.
28 (000200)	LDEBUG	0	Normal operation.
		1	Inhibit loading of test microcode.
29 (000100)	IDEBUG	0	Normal operation.
		1	Fault isolation debug mode.
30 (000040)	NIPORT	0	Do not specifically select NI port for testing.
		1	Select NI port for testing.
31 (000020)	CIPORT	0	Do not specifically select CI port for testing.
		1	Select CI port for testing.
32 (000010)	Unused		
33 (000004)	Unused		

34 (000002)	MMPROC	0 1	The MPROC module is inserted. The MPROC module is not inserted.
35 (000001)	MCBUS	0 1	The CBUS module is inserted. The CBUS module is not inserted.

7.2.2 Right Hand Switches (Detailed Description)

Switch 18 (TRACE)

This switch is used if the user wants to follow test progress. As each test is started, the diagnostic prints the name of the test and the current PC. This is useful to gain assurance that the diagnostic is running or to find out what test it died on, if the diagnostic fades into oblivion.

The names of all tests selected are printed. If the test has been disabled or if it is one the the tests that cannot be run in user mode, the test name is printed followed by the phrase "(Disabled)". Tests that run in exec mode only will be followed by the phrase "(Exec mode only)" when running in user mode.

Switch 19 (INHFLT)

If set, this switch inhibits fault isolation. All selected tests are run even if the fault isolation algorithm indicates that the test need not be run. Fault isolation data may still be obtained; however, the result is probably not be as valid.

If not set, only those tests are run which the fault isolation algorithm determines should be run.

Switch 20 (INHMSG)

This switch is used to run the diagnostic with all normal printout except error messages. It can be used in place of "NOPNT" switch when looping on an error.

Switch 21 (LOOPGM)

If this switch is not set and OPRSEL switch is not set, the diagnostic runs 5 passes of all tests. Then the diagnostic returns to DIAMON or D20MON or KLDCP.

If this switch is set and OPRSEL switch is not set, the diagnostic runs all tests continuously, pass after pass.

If OPRSEL switch is set, the diagnostic asks "What Test - " to determine what test to run. When done with a test, the diagnostic returns to this question. The LOOPGM switch is not used.

Switch 22 (LOOPTS)

This switch allows a specified test to be looped on forever. It is equivalent to specifying "EB22 37777777777" to run test EBUS test 22 a very large number of times. This switch may be overridden by specifying an explicit repeat count.

This switch is effective only when selecting a single test. If a class of tests is selected, the switch is ineffective.

If this switch is not set the default repeat count when selecting a specific test is 1 repetition.

Switch 23 (RUNALL)

This switch forces every segment of a test to be run. Normally, a test exits after the first error, unless looping on error. To force the diagnostic to run every segment of the test regardless of error, this switch may be set.

This switch has no effect if LOOPER switch is set and the diagnostic is looping on an error.

Switch 24 (DSPEAR)

This switch enables or disables SPEAR error reporting. If the switch is not set, the diagnostic will make two SPEAR entries - (1) an entry when the diagnostic is started and (2) an entry either upon first error or at end of pass if no errors have occurred.

If the switch is set, no SPEAR entries will be made. If the diagnostic has already started, the second entry will not be made.

Switch 25 (UDEBUG)

This switch allows the diagnostic to be run in user mode without a port for the purposes of diagnostic development or debug or for the purpose of familiarization with the diagnostic without affecting or requiring a port.

If this switch is set, the diagnostic runs every test as demanded. All tests and most commands in the DEBUG test appear to succeed.

Switch 26 (MDEBUG)

This switch forces every test to fail and prints out an error message for each failure. This may be used to debug error messages, or familiarize a person with typical error printouts of this diagnostic. It can also be used when trying to diagnose a failing test which passes the first few test segments - this switch causes error messages to be printed for passing portions of the test so that it can be seen what the port is doing correctly, without looking through the diagnostic listing.

Switch 27 (SDEBUG)

This switch is used for diagnostic debug purposes. If set, each table entry of a table driven test is printed as the test is executed.

Switch 28 (LDEBUG)

The first action of any test is normally to load the microcode required by the test. Setting this switch inhibits the automatic loading of microcode. This allows test microcode to be modified using the DEBUG facility and the test to be rerun.

Switch 29 (IDEBUG)

This switch is used to debug the fault isolation algorithm. If set, the possible faulty networks and probably good networks are printed at various stages during testing.

If not set, testing occurs as usual, with the usual amount of printout.

Switch 30 (NIPORT)
Switch 31 (CIPORT)

When the diagnostic determines the hardware configuration, it finds out if what ports exist and whether or not each can be selected for testing. In exec mode a port can be selected whether or not it appears to exist. In user mode the diagnostic must be able to successfully request the port from the system in order to determine that the port is selectable.

If neither switch is set, the diagnostic will ask what ports to test.

If at least one of the switches is set, the diagnostic will select the port or ports specified whether or not it appears to exist, unless the port cannot be selected (user mode only). The diagnostic will not ask the use whether or not the port should be tested.

Switch 32 - Unused

Switch 33 - Unused

Switch 34 - MMPROC

This switch, when set, is used to indicate to the diagnostic that the Microprocessor control module is not inserted. No tests which require use of that module are run.

If this switch is not set, the diagnostic runs any test which requires at least an EBUS and MPROC module, but not a CBUS module. If neither this switch nor MCBUS switch is set, tests using any module may be run.

Switch 35 - MCBUS

This switch, when set, is used to indicate to the diagnostic that the CBUS module is not inserted. No tests which require use of that module are run.

If this switch is not set, the diagnostic runs all tests, depending only on the state of Switch 34.

8.0 ERRORS

This section describes the operation of the diagnostic upon encountering various types of errors. There are basically three types:

- User input errors
- Hardware errors
- Unexpected errors

8.1 User Input Errors at the Terminal

The diagnostic requests three types of input: numerical, yes or no, and SIXBIT. The action taken upon input error is as follows:

When an error is detected on numerical input the diagnostic prints an error message describing the type of error. Then it either asks the question again, or returns to "What test -" or "DEBUG>".

When an error is detected on a yes or no input, the user is merely reprompted, until a Y or N is typed.

When an error is detected on SIXBIT input an error message is printed and the user is asked the question again. The user can then ask the diagnostic for further information by asking for help or typing "?".

8.2 Error Reporting of Hardware Detected Errors

Upon the occurrence of a hardware error a full error description is printed, depending on the setting of switches "NOPNT", "PNTLPT", "TXTINH", and "INHMSG".

If the diagnostic is looping on error only the first occurrence of an error is printed. Subsequent errors at the same place are not printed. If the "DING" switch is set, the terminal bell is sounded to indicate that an error was detected, even though nothing is printed.

8.2.1 Reporting to SPEAR

The diagnostic makes 2 SPEAR entries. The first is an entry when the diagnostic is started. This entry contains the date, time, user, diagnostic name and version. A second entry is made when the first error occurs. This entry contains the date, time, user, diagnostic name and version, failing test name and test description.

If the diagnostic completes a successful pass with no errors, it will make the second entry as above, but there will be no test name or test description included.

If, thereafter, an error occurs and the diagnostic has already made an entry indicating successful completion of a pass, the diagnostic will make one more entry, giving the error information.

Whenever the diagnostic is restarted by an STM or reloaded and restarted, it will make entries into the system error file. If the 2 entries have already been made, restarting via a STD or CONT will not result in additional entries.

SPEAR entries can be disabled using the switch "DSPEAR".

8.2.2 Error Printouts

Errors are printed as they occur in a standard form. Where appropriate, the CSR register or LAR register contents is printed. A sample printout with TXTINH switch not set follows:

```
+-+-+-+-+-+-+
? CI Error:  TEST CB1-1 - Fmtr Data Loopback
Test PC = 222"  Scope PC = 237"  Switches = 020010 000000
Data loopback through Fmtr failed (result in EBUF)
Start Addr:  0000 , # Steps:  6.
End Addr - Correct:  5004
             Actual:  4002
EBUF - Correct:  777777 777777
        Actual:  000000 230014
-------------
```

This error printout gives the following data:

- Port indication - CI.
- Test name - CB1-1 (CBUS test #1 segment #1), name of test is "Fmtr Data Loopback".
- Test PC - 222" - address of the start of the test. Note that this is a relative address as will be found in the test listing.
- Scope PC - 237" - address which can be found in the diagnostic listing of the call to the scope loop routine. Note that this is a relative address also.
- Switches - those in effect at the time of the error printout.
- "Data loopback through Fmtr failed (result in EBUF)" - description of error.

- Test related information such as
  - Starting CRAM address - 0000
  - Number of single steps done - 6.
- Error information such as
  - Where the test should have stopped (CRAM address 5004) and where it actually did (CRAM address 4002)
  - EBUF data correct and actual

If TXTINH is set, the printout looks like:

```
+-+-+-+-+-+-+
? CI Error:  TEST CB1-1  Scope PC 237"  SW 020210 003020
Start  0000 , # Steps  6.
End (C):  5004
    (A):  0000
EBUF (C):  777777 777777
     (A):  000000 230014
-------------
```

8.3 Unexpected Errors

The diagnostic is written to recover from any usual error condition and continue testing. However, there are a few halts coded into the diagnostic to catch a completely lost program because of some unexpected malfunction:

ADDRESS	REASON
1011	Fatal push list pointer error
1012	Initial push list pointer error
1013	MUUO with LUUO handler wiped out
1014	DTE20 interrupt without doorbell
1015	DTE20 clock interrupt without flag set
1016	CPU initialization error
1017	End of program error
1020	Interrupt with LUUO handler wiped out

The diagnostic is coded with several fatal instructions which cause

Fatal program error at address xxxxx

to be printed and the diagnostic to halt. These instructions are placed where the diagnostic finds itself in a situation where it does not know how to proceed. There are no anticipated program deficiencies that allow a fatal error to be executed.

9.0 PROGRAM DESCRIPTION

This section provides a functional overview of the diagnostic, giving information concerning requirements and restrictions, fault detection and isolation, error handling, terminal input and output characteristics, and diagnostic applications.

9.1 Program Overview

9.1.1 Program Purpose

DFPTA is a basic device diagnostic intended to detect and isolate hard or stuck at faults for the CI20/NIA20 controller on the KL10. It tests only CI20/NIA20 controllers consisting of 3 port modules located in RH20 slots #5 and #7.

9.1.2 Testing Strategy

The diagnostic attempts to run simpler tests first that exercise a small amount of logic. Then more and more testing is done, until the tests are exercising much of the logic of the port at once. First, the EBUS interface is tested, then other portions of the EBUS module. Then, the data path used in loading microcode is tested. Then, microcode is loaded and the 2910 sequencer is tested. Then, more of the EBUS module is tested including the 2901s. Then, more of the MPROC module. Then, the CBUS internal data path, followed by the channel interface, followed by the rest of the CBUS module.

If test tracing is being done, it will be noted that the tests are not run in numerical sequence.

If fault isolation is disabled (the INHFLT switch is set), the diagnostic runs each test in numerical sequence - EBUS tests, then MPROC tests, then CBUS tests.

9.2 Run-Time Dynamics

9.2.1 Memory Allocation Requirements

The diagnostic loads and executes in less than 96K of memory. This includes the memory used by the diagnostic monitor (DIAMON or D20MON or MAGMON) and the subroutine package (SUBRTN or SUBUSR or SUBKL).

9.2.2 Operating System Interactions

9.2.2.1 TOPS-10

The diagnostic runs under TOPS-10 release 7.03 and later using the DIAG. UUO. The diagnostic requests and releases the port using the DIAG. UUO.

The diagnostic can only be run if the port is not being used by the operating system. This is done in a single CPU environment by dismounting any mounted structures on all HSC50s. In a multiple CPU environment, other CPUs will have access to the disks attached to HSC50s using their CI20 ports, so the structures need not be dismounted.

The CI port is made available for diagnostic use by the OPR program's "CONFIG REMOVE CI-n" command, where "n" is the CPU number.

9.2.2.2 TOPS-20

The diagnostic interfaces with TOPS-20 via the DIAG JSYS. The diagnostic requests the port using the DIAG JSYS. When done, the diagnostic releases the device also using the DIAG JSYS.

The diagnostic can only be run if the port is not being used by the operating system. The DIAG JSYS will succeed only if this is so. Hence, the user must close all CI connections before running this diagnostic.

On TOPS-20 the CI is used for data transfer with HSC50 based disks and for communication with other TOPS-20 systems. Communication over the CI is required for operation under CFS-20 (TOPS-20 Common File System). Under TOPS-20 release 6.1 and later, Phase IV DECnet-20 communications to other TOPS-20 systems may be active over the CI.

The CI is removed from operating system use by dismounting all mounted structures on any HSC50 and closing all other communications over the CI. This is accomplished via the OPR program's "SET PORT CI UNAVAILABLE" command.

9.2.3 User Mode Restrictions

None.

9.2.3.1 User Mode Privileges

Under TOPS-20 a user must have at least MAINTENANCE privileges in order to run the diagnostic. Users with WHEEL or OPERATOR capability also have sufficient privileges to run the diagnostic.

Under TOPS-10, you must be running as [1,2] or have privileges to execute the TRPSET and DIAG. UUOs.

9.2.3.2 User Mode Priority Interrupts

The diagnostic is not allowed to use the Priority Interrupt System in user mode. In user mode, the diagnostic polls the CSR Register for interrupt conditions.

9.3 Exec Mode Capability

The diagnostic runs in exec mode under KLDCP or, additionally, under DIAMON or MAGMON. All tests are capable of being run in exec mode.

9.4 User Mode Capability

The diagnostic runs in user mode under TOPS-20 (release 6.0 or newer) and TOPS-10 (release 7.03 or newer). There are no restrictions imposed upon the diagnostic in user mode.

9.5 Fault Detection

9.5.1 Error Resolution

The diagnostic isolates faults to a network of failing chips. Each network is a set of chips, generally functionally related. Typically, several networks are printed, with the first network being the most probable.

9.5.2 Fault Detection And Isolation

The purpose of the diagnostic is to diagnose hard faults on the CI20/NIA20 Controller and to isolate those faults to a specific failing field replaceable unit (FRU). The diagnostic testing is performed by a bottom up testing technique. This is accomplished by performing a basic operation and then using that verified logic to verify the functionality of more logic.

The diagnostic should detect 99% of solid faults in the port modules and isolate 95% of these to the FRU.

9.5.3 Fault Detection

Typically, the diagnostic detects a fault, then prints it out and continues testing. When done the failing module is called out followed by a list of failing networks.

Occasionally, particularly with EBUS interface failures, the diagnostic may just hang, or the KL10 microcode may hang. This may happen if the port cannot complete an EBUS operation such as an examine or deposit properly. If this occurs, no error message or module callout is printed. The solution is to restart the diagnostic and run the tests over again with the test tracing switch, "TRACE", set. Then, the failing test can be observed. From this point, the diagnostic listing for the test should be inspected to see what is being done. If the port was executing microcode at this time, the LAR can be read and microcode single stepped to duplicate the failure.

9.6 Network Descriptions

9.6.1 EBUS Module networks

9.6.1.1 E1 - EBUS Control Logic

This consists of most of the control logic on page EBI1 which does DATAI, DATAO, CONI, CONO decoding, IOP decoding, and interrupt handling.

9.6.1.2 E2 - CC EBUS RQST

This is located on page EBI1 and consists of the control logic driving the condition code "EBI1 CCEBUSRQST".

9.6.1.3 E3 - CC GRNT CSR

This is located on page EBI1 and consists of the control logic driving the condition code "EBI1 CCGRNTCSR" and several related signals "EBI1 MPRQSTCSRLTCH" and "EBI1 PORTGRNTCSR".

9.6.1.4 E4 - Interrupt Logic

This is located on page EBI2, labelled "Interrupt Generation Logic". Also included is two related pieces of logic on the same page which drive the signals "EBI2 LOADEBUS" and "EBI2 XFER".

9.6.1.5 E5 - Clock Logic

This consists of the inverters on page EBI2 from which come clocks 1 and 3.

9.6.1.6 E6 - Clear Logic

This is located on page EBI2, and consists of the logic driving the signals "EBI2 CLREBUF", "EBI2 CLEARA" and "EBI2 CLEARB". Also included is a bit of logic on page EBI4 which determines "EBI4 CSR18".

9.6.1.7 E7 - Magic # Field Decode

This is located on page EBI2.

9.6.1.8 E8 - CC CSR CHNG

This is located on page EBI2, and consists of the logic which drives the condition code "EBI2 CCCSRCHNG".

9.6.1.9 E9 - KMUX

This is located on page EBI3, labelled "KMUX". Also included is the enable logic on page EBI2, "EBI2 ENAKMUX", "EBI2 EBPAROUT", and "EBI2 EBPARACT".

9.6.1.10 E10 - EMUX

This is located on page EBI3, labelled "EMUX". Also included is the enable on page EBI2, "EBI2 ENAEMUX".

9.6.1.11 E11 - RMUX/CSR 25-31

This is located on page EBI4, and consists of the control logic driving CSR bits 25-31 and the RMUX (labelled appropriately). There is also a gate on page EBI2 which is included, which is 'EBI2 RMUXLOADCSR''.

9.6.1.12 E12 - CSR 11-13/19-22/33-35

This is located on page EBI4, and consists of the control logic driving CSR bits 11-13, 19-23 and 33-35.

9.6.1.13 E13 - CSR 04 Rqst Exam/Dep

This is located on page EBI4, and consists of the control logic driving the signal 'EBI4 CSR04''.

9.6.1.14 E14 - CSR 05 Rqst Interrupt/CC INTR ACTIVE

This is located on page EBI4, and consists of the control logic driving the signals 'EBI4 CSR05'', 'EBI4 MPROCERR'', 'EBI4 CLRRUN'', and 'EBI4 CCINTRACTIVE''.

9.6.1.15 E15 - CSR 06/07 CRAM PE/MBUS Error

This is located on page EBI4, and consists of the control logic driving the signals 'EBI4 CSR06'' and 'EBI4 CSR07''.

9.6.1.16 E16 - CSR 24 EBUS PE

This is located on page EBI4, and consists of the control logic driving the signals 'EBI4 CSR24'' and 'EBI4 CCEBPARERR''.

9.6.1.17 E17 - CSR 32 MPROC Run

This is located on page EBI4, and consists of the control logic driving the signal 'EBI4 CSR32''.

9.6.1.18 E18 - KMUX Parity Generation

This is located on page EBI5, and consists of 5 parity generators, the output of which is "EBI5 KMUXPAR H".

9.6.1.19 E19 - EBUS Parity Checking

This is located on page EBI5, and consists of 5 parity generators, the output of which is "EBI5 EBPARERR H".

9.6.1.20 E20 - EBUF

This is located on page EBI5, labelled "EBUF".

9.6.1.21 E21 - EBUS Data Xcvrs

This is located on the top half of page EBI6, and consists all of the EBUS transceivers carrying data signals.

9.6.1.22 E22 - EBUS Control Xcvrs

This is located on the bottom half of page EBI6, and consists all of the EBUS transceivers carrying control signals.

9.6.1.23 E23 - 2901s/Carry Logic

This occupies all of pages EBI7 and EBI8.

9.6.1.24 E24 - Constant MUX

This is located on page EBI9, labelled "CNST MUX".

9.6.2 MPROC Module networks

9.6.2.1 M1 - MPROC Run Clocks

This is located on page MPR1, labelled "uPROC Run Clocks". Also included are the 'MPR1 CLK1A, CLK2A, CLK3A, CLK4A" also on page MPR1. Also, 'MPR1 CSR32" and 'MPR1 Error" are grouped in this category.

9.6.2.2 M2 - Local Memory Control

This is located on page MPR1, labelled "Local Memory Control".

9.6.2.3 M3 - MBUS Error Detect Logic

This is located on page MPR2, labelled 'MBUS Error Detect Logic". Also included are two inverters on page MPR1 'MPR1 EnaEMUX" and 'MPR1 EnaCMUX".

9.6.2.4 M4 - CRAM Read/Write Control

This is located on page MPR2, labelled "CRAM Read/Write Control". Also included is some inverters on MPR1 which fan out some of the cram control signals.

9.6.2.5 M5 - Jump ADDR MUX

This is located on page MPR3, labelled "Jmp MUX".

9.6.2.6 M6 - CC MUX

This is located on page MPR3, labelled "CC MUX".

9.6.2.7 M7 - 2910 MicroSequencer

This is located on page MPR3, labelled 'MICROSEQR".

9.6.2.8 M8 - CRAM ADDR MUX

This is located on page MPR3, labelled "Addr MUX".

9.6.2.9 M9 - LAR

This is located on page MPR3, labelled "LAR". Loading the LAR requires the signal "MPR1 LOADLAR" which is located on page MPR1. The logic involved with this is included in this category.

9.6.2.10 M10 - CRAM 0000-7777

This consists of all of the 4k x 4 rams on pages MPR4 and MPR5.

9.6.2.11 M11 - Cond/Skip Decoder

This is located on page MPR6, labelled "Cond/Skip Decoder".

9.6.2.12 M12 - CRAM Control Register

This occupies all of page MPR7, labelled "CRAM Ctrl Reg". Also included here are the parity checkers for this register, located on page MPR6, labelled "CRAM Par Chkr".

9.6.2.13 M13 - Local Storage RAM

This is located on page MPR8, labelled "Local Store".

9.6.2.14 M14 - Local Storage Addr Register

This is located on page MPR8, labelled "Loc Stor Addr Reg". In addition, the signal "MPR1 CLKSADREG" is included, on page MPR1.

9.6.2.15 M15 - RAM Mode MUX

This is located on page MPR8, labelled "RAM Mode MUX".

9.6.2.16 M16 - RAR

This is located on page MPR9, labelled "RAR".

9.6.2.17 M17 - Right CRAM Load Buffer

This is located on page MPR9, labelled "Right CRAM Load Buff".

9.6.2.18 M18 - Left CRAM Load Buffer

This is located on page MPR9, labelled "Left CRAM Load Buff".

9.6.2.19 M19 - MW Out MUX

This is located on page MPRA, labelled "MW Out MUX".

9.6.3 CBUS Module networks

9.6.3.1 C1 - CBUS Control Logic

This consists of the logic which controls the operation of the channel, including the logic which cycles through the select, request, wait, and data states and the logic which handles the "MPSTORECBUS" and "MPWRITEMEM" signals. This is found on pages CBI1, CBI2, and CBI5.

9.6.3.2 C2 - Clock Logic

This is located on page CBI1, labelled "Clock Generation Logic".

9.6.3.3 C3 - Reset Logic

This is found on page CBI1, a group of inverters with the output signal "CBI1 Reset L".

9.6.3.4 C4 - Clear Logic

This is found on page CBI2, two inverters and two nand gates with the input signal "MPR1 ClearA L".

9.6.3.5 C5 - "CC CBUS AVAIL"

This consists of the logic which implements this condition code and it is located on page CBI2.

9.6.3.6 C6 - "CC CB LST WD"/ "CC CHAN ERR"

This consists of the logic which implements these condition codes and it is located on page CBI2.

9.6.3.7 C7 - "CLR CC CODE"

This is found on page CBI2, having the output signal "CBI2 ClrCCCode L".

9.6.3.8 C8 - Magic # Field Decode

The magic number field is decoded on page CBI3. This logic occupies most of this page. A few inverters are also shown on page CBI4 - "CBI4 EnaCMUXA" and "CBI4 PlinToCMUXA".

9.6.3.9 C9 - "CC CBUS PAR ERR"

This consists of the logic which implements this condition code and it is located on page CBI3.

9.6.3.10 C10 - "CC PLI PAR ERR"

This consists of the logic which implements this condition code and it is located on page CBI3.

9.6.3.11 C11 - Parity Predictor

This is located on page CBI4, labelled "Parity Predictor".

9.6.3.12 C12 - Formatter Control

This is located on page CBI4, labelled "Formatter Control".

9.6.3.13 C13 - PLI Control Logic

This is located on page CBI4, labelled "PLI Ctrl Logic".

9.6.3.14 C14 - CBUS Input Buffer

This is located on page CBI5, labelled "CBUS Input Buffer".

9.6.3.15 C15 - CBUS Output Buffer

This is located on page CBI5, labelled "CBUS Output Buffer".

9.6.3.16 C16 - CBUS Out Par Gen

This is located on page CBI6, labelled "CBUS Out Par Gen".

9.6.3.17 C17 - CBUS In Par Checker

This is located on page CBI6, labelled "CBUS In Par Checker".

9.6.3.18 C18 - DMUX

This is located on page CBI7, labelled "DMUX".

9.6.3.19 C19 - SMUX

This is located on page CBI7, labelled "SMUX".

9.6.3.20 C20 - Mvr/Fmtr

This is located on page CBI8, labelled "Mvr/Fmtr".

9.6.3.21 C21 - PLI Logic

This occupies most of page CBI9, labelled "PLI Par Input Checker", "PLI Par Output Generator", and "PLI Output Buffer".

9.6.3.22 C22 - PMUX

This is located on page CBI9, labelled "PMUX".

9.6.3.23 C23 - CMUX

This is located on page CBIA, labelled "CMUX".

9.6.3.24 C24 - CBUF

This is located on page CBIA, labelled "CBUF".

9.7 Performance During Hardware Failures

9.7.1 CPU/Memory Failures

Most CPU and memory faults are fatal errors to the diagnostic and it does not continue after one.

9.7.2 Power Fail/Restart

If a power fail occurs, the diagnostic must be reloaded and rerun when the system is back in operation.

9.7.3 Port Failure

Hardware failure in the port is anticipated and results in an error message.

9.7.4 Unexpected Traps

The diagnostic can fail due to an unexpected malfunction. Sometimes an error code is printed, if the subroutine package intercepts the error. Other cases may result in a hung diagnostic or CPU or occasionally a fatal error message. See the Section 8.3 "Unexpected Errors" for more details.

9.8 Program Applications

The diagnostic can be used for coarse determination of a fault - does the port work and if not what module is broken.

It can also be used for detailed examination of a port.

9.8.1 Users and Uses

The diagnostic is generally used only by Manufacturing to assist module construction and debug, and by Customer Services when determining the FRU of a failing port.

9.8.2 Engineering Usage

Engineering used this diagnostic for breadboard debug, design verification, and for prototype debug and maintenance. Engineering may use the diagnostic if any ECOs are made in the future.

9.8.3 Manufacturing Usage

This diagnostic is used by Manufacturing as part of the acceptance procedure for an CI20/NIA20. It is also used to repair faulty port modules (EBus Interface, Micro-Processor Control, and CBus Interface modules).

9.8.4 Customer Services Usage

Customer Services uses the diagnostic as part of acceptance procedure, when an CI20/NIA20 is sent to a customer in a drop ship environment. Customer Services also uses this during CM and PM.

9.8.5 Customer Usage

With the exception of OEM customers, it is not likely that any customers will use the diagnostic. If they have a problem with the port they notify Customer Services who can then run the diagnostic. OEM customers, running the diagnostic themselves, would interface to their account sales representative if any difficulties arose or problems discovered.

9.9 Test Design

The basic design of most tests is as follows:

- o Do something to the port.
- o Inspect the results.
- o If error, print an error message.

- If error and error looping enabled, loop on error.

This fits into the diagnostic as described by the following steps done by the diagnostic:

- Do program initialization
- Select test category
- Run these tests
  - Select a test
  - Run it
  - Select next test, exit if no more
- Select next test category to test if OPRSEL switch is set, otherwise exit the diagnostic.

9.9.1 Type of Tests

There are three types of tests:

- Hardware Test
- Program Option Selection
- Debugger Test

Whenever the tests are run without manual control of which test is selected, only hardware tests are run. This is the major function of the diagnostic and these tests comprise 95% of the diagnostic.

Program option selection tests can be selected when manual control of what test to run has been selected. These are not tests in that they do not do anything to the hardware. But since they are selected from a list of hardware tests, they are called tests. These consist of switch selections or changes, help, and exit commands.

The debugger test is a special test that allows one to to CONI, CONO, DATAI, and DATAOs to the port and in effect write ones own test. The purpose of this test is for hardware debug of the port, where one is interested in doing things and inspecting the results in a less formal fashion than just running tests. This is only one test but it allows many different things to be done to the port. It is intended to be used primarily in breadboard debug, but may be used later by Customer Services or Manufacturing.

9.9.2 Test Flow

If only one port is being tested, all of the tests specified are run on the port. Then fault isolation data is printed if any errors occurred.

If both NI and CI ports are being tested simultaneously, the diagnostic runs all of the tests on the NI first, printing fault isolation data afterwards. Then all of the tests are run on the CI following by printing fault isolation data.

The following sections describe starting and stopping tests.

9.9.2.1 Starting the Test

After the program switches have either been given via console switches or by typing them in on the terminal, the diagnostic starts the tests as selected by the switches or by user input.

If test tracing is enabled, the diagnostic prints the test pc and test name at the beginning of each test, and a phrase such as "(Disabled)" or "(Exec mode only)" if the test actually is not going to be run.

A test is disabled if such has been specified with the DISABLE test command. A test is inhibited only in user mode where the test cannot be run because it uses the PI system or causes page faults. There are about 15 of these tests.

9.9.2.2 Stopping the Test

A test sequence may stop either by terminating normally, by halting on error (if the ERSTOP switch is set), by the user setting the ABORT or RSTART switches, or by typing an Altmode or C. In user mode, ABORT and RSTART are ignored if set.

The diagnostic continues at one of several places depending on switch settings and whether or not the diagnostic is being run in exec or user mode and what loader has been used to load the diagnostic:

- o D20MON CMD -
- o DIAMON CMD -
- o MAGMON CMD -

- What Test -
- . (TOPS-10 prompt)
- @ (TOPS-20 prompt)
- o >. (KLDCP prompt)

9.9.2.3 Obtaining Run Time Status (Control-T)

In user mode the current program status is given when T is typed. This is not available in exec mode.

9.9.2.4 Obtaining Run Time Status (S or Control-S)

In exec mode, while running tests, the current test name and time is printed when an S <CR>, s <CR> or Control-S is typed.

In user mode, typing an S or s produces the printout.

10.0 TEST DESCRIPTIONS

10.1 Test Description Assumptions

All test descriptions assume that no program switches are set. The action of a test if a particular switch is set may be determined by reviewing the function of the switch, given in prior sections.

10.2 Test Description Overview

The description for "program option selection" test consists of what each item does and a description of its use.

The description given for a "hardware" test consists of a single line test description. The diagnostic listing provides a detailed description at the beginning of each test. Specific implementation details can be obtained from the listing or from the microcode of the test.

The description for the "debug" test gives a list of commands and what each does.

10.3 Program Option Selections

10.3.1 Basic Description

This is a set of commands that can be given to facilitate running tests or debugging the port.

10.3.2 Commands Available

The commands available in regular mode are as follows, which are typed in response to the prompt 'What test - ':

Hardware tests:

NAME	DESCRIPTION
ALL n	All tests (n times)
EBUS n	EBus module tests
SEQ n	SEQ related tests
ALU n	2901 related tests
MPROC n	MPROC module tests
CBUS n	CBus module tests
MBUS n	MBUS related tests
XXm n	Test m, category XX (where XX can be EB,SE,AL,MP,CB)

General Commands:

NAME	DESCRIPTION
HELP	Print this message
EXIT	Return to normal test dispatching
DDT	Enter DDT (Type RTN$X to return)
TAKE	Take cmds from file
NTAKE	Take cmds from file (not echoed)
DEBUG	Enter port debugger
DIAMON	Enter DIAMON

Test Control Commands:

NAME	DESCRIPTION
ENABLE n	Enable execution of test n
DISABL n	Disable execution of test n
DISLIS	List disabled tests
LIST n	List all tests (or just test n)
LISTEB n	List EBus module tests (or test n)
LISTSE n	List Sequencer related tests
LISTAL n	List 2901 related tests
LISTMP n	List MPROC module tests
LISTCB n	List CBus module tests

Device Selection Commands:

CONFIG	Determine/print configuration
CONPNT	Print configuration
SELECT	Select device to test

Switches:

SWITCH	Enter switch information
SWPRIN	Print current switch settings

NOPNT	Complement "no print" switch
PNTLPT	Complement "print lpt" switch
DING	Complement "bell on error" switch
LOOPER	Complement "loop on error" switch
ERSTOP	Complement "halt on error" switch
PALERS	Complement "print all errors" switch
RELIAB	Complement "reliability mode" switch
TXTINH	Complement "text inhibit" switch
OPRSEL	Complement "operator select" switch

TRACE	Complement "trace test" switch
INHFLT	Complement "inhibit fault isolation" switch
INHMSG	Complement "inhibit error message" switch
LOOPGM	Complement "loop on pgm" switch
LOOPTS	Complement "loop on test" switch
RUNALL	Complement "run all test segments" switch
DSPEAR	Complement "disable SPEAR reporting" switch
UDEBUG	Complement "user mode debug" switch
MDEBUG	Complement "error message debug" switch
SDEBUG	Complement "test debug" switch
LDEBUG	Complement "load microcode debug" switch
IDEBUG	Complement "fault isolation debug" switch
NIPORT	Complement "Select NI port" switch
CIPORT	Complement "Select CI port" switch
MPROC	Complement "missing MPROC module" switch
MCBUS	Complement "missing CBUS module" switch

10.3.3 Detailed Command Description

In more detail these commands are as follows:

10.3.3.1 Hardware Tests

ALL n

This command will run all hardware tests the number of times specified by the repeat count. If no repeat count is specified, the diagnostic asks how many passes to run.

If fault isolation is not inhibited, fault isolation data will be printed upon completion of the tests. If testing is aborted by typing an altmode, the fault isolation data is printed at that point.

EBUS n

This command will run all EBUS module tests in the same manner as ALL above. This includes the 2901 ALU testing.

SEQ n

This command will run all 2910 micro-sequencer tests in the same manner as ALL above.

ALU n

This command will run all 2901 ALU tests in the same manner as ALL above.

MPROC n

This command will run all MPROC module tests in the same manner as ALL above.

CBUS n

This command will run all CBUS module tests in the same manner as ALL above.

MBUS n

This command will run all MBUS cable related tests in the same manner as ALL above. The diagnostic also prints a short description of how to debug an MBUS problem. This description follows:

```
To isolate a possible MBUS problem:

 (1) Disconnect MBUS cable from EBUS module.
     Run MBUS tests with MMPROC and MCBUS switches set.
     Failed - EBUS module is at fault.  Stop.
     Passed - MBUS cable or MPROC or CBUS modules at fault.
              Continue at next step to isolate further.

 (2) Connect MBUS cable to EBUS module only.
     Run MBUS tests with MMPROC and MCBUS switches set.
     Failed - MBUS cable is at fault.  Stop.
     Passed - MBUS cable or MPROC or CBUS modules at fault.
              Continue at next step to isolate further.

 (3) Connect MBUS cable to EBUS and MPROC modules only.
     Run MBUS tests with MCBUS switch set.
     Failed - MPROC module is at fault.  Stop.
     Passed - CBUS module at fault.  Stop.
```

XXm n

This command is given to run a single test. The test name is given as a 1 or 2 letter test category followed by the test number. The test categories and sample test names are as follows:

o Ebus module test ..... EB10 or E10

o 2910 Sequencer test .. SE10 or S10

o 2901 ALU test .... AL10 or A10

o MPROC module test .... MP10 or M10

o CBUS module test ..... CB10 or C10

A repeat count may be given. If not given, the test is executed only once.

10.3.3.2 General Commands

HELP

This command prints a one line description for each command.

EXIT

This command returns control to the diagnostic loader or to KLDCP if no loader program was used to load the diagnostic in exec mode.

DDT

This command invokes DDT. When RTN$X (POPJ 17,0) is executed, the diagnostic resets the PI system in exec mode and returns to 'What Test - " question. DDT can also be entered by typing Control-D and when done typing RTN$X. When invoked in this fashion, the diagnostic does not reset the PI system.

TAKE file ext

This command takes commands from a file. The file may be specified as an argument to the command. If no file name is specified, the default file name is used. The default file name is either 'DFPTA.CMD" or the last file specified.

Any legal command may be specified except another TAKE or commands which cause exit from the diagnostic such as DDT or DIAMON or EXIT. DIAMON or EXIT may be included but processing of the take file will terminate with the command. The diagnostic prints each command and result on the terminal as it executes each command.

The file specified must be on the load device known to MAGMON or D20MON or DIAMON. If not, an error message is printed and the take command is aborted.

If any error occurs while processing the take file, the take command is aborted. Errors include any unexpected errors such as unrecognized command, failures to start the port, argument errors, etc.

NTAKE file ext

This command is identical to the TAKE command with the single exception that the commands executed are not echoed on the terminal. All output resulting from execution of the commands is echoed.

DEBUG

Enter port debugger. This test allows specific operations to be done with the port without the restrictions of executing a particular test.

DIAMON

Enter DIAMON. This is the same as the EXIT command if a diagnostic loader is present. If MAGMON is loaded instead of DIAMON, control returns to MAGMON.

10.3.3.3 Test Control Commands

ENABLE n,n,n,... and DISABL n,n,n,...

The running of a particular test can be explicitly prevented with the DISABL command. The ENABLE command allows the execution of the specified tests to be enabled again. Disabling a test will guarantee that the test will not be run. Enabling a test will allow a test to be run, but whether or not it is run, depends on the determination made by the fault isolation algorithm and the availability of the required port modules.

Test names are given as "Test Class" "Test Number". where test class is:

- SE or S - 2910 sequencer tests
- MP or M - Mproc module tests
- AL or A - 2901 ALU tests
- CB or C - CBus module tests
- EB or E - EBus module tests

Sample test names are SE1, SE27, EB1, E1, AL20, ...

Example:

```
What test - ENABLE EB3,EB4
What test - DISABL EB3,EB4,S2,SE70,AL10,A11
```

DISLIS

This command lists what tests have been disabled. Example:

```
What test - DISLIS

EB1 EB3 EB4 SE1 SE2 MP1 MP2
```

LIST n

List all tests (or just test n). This command prints the test description which is also printed in each error message. If no argument is given, all test descriptions are printed (this can be aborted by typing an altmode). If a test name such as MP2 is typed, only the description for that test will be printed. Example:

```
What test - LIST EB2

EB2 - RH20/Port Interaction

What test - LIST

EB1 - CONI Works?
EB2 - RH20/Port Interaction
EB3 - RH20/Port Interaction
...
```

LISTxx n,n

Test descriptions can be selected for printing within a class. A command of the form "LIST" followed by the "test class" code specifies tests within the test class. The commands are as follows:

- LISTEB n,n - List EBus module tests (or test n)
- LISTSE n,n - List Sequencer related tests
- LISTAL n,n - List 2901 related tests
- LISTMP n,n - List MPROC module tests
- LISTCB n,n - List CBus module tests

Up to 2 numeric arguments may be given which are decoded as follows:

- o No arguments - all tests are listed.
- o One argument - such as 'LISTSE 4' - list only the specified test (in this example it is test SE4).
- o Two arguments - such as 'LISTSE 4,10' - list tests 4 through 10 in the selected test class.

Example:

```
What test - LISTSE 4,5

SE4 - JZ Instruction - Effect on Reg/Ctr
SE5 - CONT Instruction

What test - LISTMP

MP1 - CRAM Data Test RAR12=0
MP2 - CRAM Data Test RAR12=1
...
```

10.3.3.4 Switches

SWITCH

Enter switch information. Switches can be entered as in the following example. Usually, though, it is easier just to type the switch name to turn on or off a switch.

```
What Test - SWITCH

Print the selectable pgm switches? (Y or N) - Y

----Left Side Switches--------     ----Right Side Switches--------

SW   KL10     11     Function      SW   Value   Function
--   ----    ----    --------      --   -----   --------
 0  400000  100000   Abort         18  400000   Program trace
 ...

Switches = 020010 004000

20010  LH Switches <# or ?> - ?

Type  Y,N,^,CR,^Z

N ABORT -
N RSTART -
...
```

```
4000  RH Switches <# or ?> - ?

Type  Y,N,^,CR,^Z

N TRACE -
N INHFLT -
...
```

SWPRIN

Print current switch settings. Example:

```
What Test - SWPRIN

Switches:  020010 002000   PNTLPT OPRSEL UDEBUG
```

```
NOPNT    RELIAB   TRACE    DSPEAR   UDEBUG
PNTLPT   TXTINH   INHFLT   NIPORT   MDEBUG
DING     OPRSEL   INHMSG   CIPORT   SDEBUG
LOOPER            LOOPGM   MPROC    LDEBUG
ERSTOP            LOOPTS   MCBUS
PALERS            RUNALL
```

The name of each of these switches can be typed. The result is to complement the switch setting. If left hand switches are typed in exec mode an error message is printed.

10.4 Hardware Tests

More information on each of the following hardware tests listed here can be found in the program listing.

10.4.1 Hardware Tests - EBUS Module

10.4.1.1 EBUS Interface Tests

EB1 - CONI Works?
EB2 - RH20/Port Interaction
EB3 - RH20/Port Interaction
EB4 - RH20/Port Interaction
EB5 - RH20/Port Interaction

10.4.1.2 CSR Register Tests

EB6 - CSR Cleared After Reset
EB7 - Read/Write CSR Bits
EB10 - CSR Bits Interaction

10.4.1.3 EBUF Tests (MBus Loopback)

EB11 - EBUF Access Test
EB12 - EBUF/CSR Conflict Test
EB13 - KMUX Parity Generators
EB14 - EBUF/CSR Conflict Test
EB15 - EBUF Reset Test
EB16 - EBUF Data Path Test

10.4.1.4 CSR Bit 18 "Clear Port" Tests

EB17 - CSR Bit 18 Self Clearing Test
EB20 - CSR Bit 18 Reset Test
EB21 - CSR Bit 18 Reset Test

10.4.1.5 'Diag Test Ebuf''/Initialized

EB22 - Read/Write EBUF When Ucode Running

10.4.1.6 Parity Generators/Checkers Tests

EB23 - Par Gen - Not Forcing an Error
EB24 - Par Gen - Forcing an Error

10.4.1.7 CSR Read Only Bits

EB25 - Read only CSR Bits

10.4.1.8 CSR 01 ''DIAG RQST CSR''

EB26 - CSR 01 ''DIAG RQST CSR''
EB27 - CSR 01 ''DIAG RQST CSR''
EB30 - CSR 01 ''DIAG RQST CSR''
EB31 - CSR 01 ''DIAG RQST CSR''
EB32 - CSR 01 ''DIAG RQST CSR''
EB33 - CSR 01 ''DIAG RQST CSR''

10.4.1.9 CSR 02 ''DIAG CSR CHNG''

EB34 - CSR 02 ''DIAG CSR CHNG''

10.4.1.10 Contant MUX

EB35 - Constant MUX
EB36 - Constant MUX Interference
EB37 - CSR Bit 24 - EBUS PE Occurs at Wrong Time
EB40 - EBUS Transfer via DATAOs
EB41 - EBUS Transfer via DATAIs

10.4.1.11 CSR 05 ''Rqst Int'' Bit

EB42 - CSR 05 ''Rqst Interrupt''
EB43 - CSR 05 ''Rqst Interrupt''
EB44 - CSR 05 ''Rqst Interrupt''

10.4.1.12 CSR 06 ''CRAM PE'' Bit

EB45 - CSR 06 ''CRAM Par Err''

10.4.1.13 CSR 07 "MBUS Err" Bit

EB46 - CSR 07 "MBUS Err"

10.4.1.14 CSR Bits 11,12,13

EB47 - CSR 11 "Idle" Bit
EB50 - CSR 12 "Disable Complete" Bit
EB51 - CSR 13 "Enable Complete" Bit

10.4.1.15 MPROC Run Bit

EB53 - MPROC Run Bit
EB54 - Basic KMUX Parity Test
EB55 - Clearing MPROC Run Bit
EB56 - Clearing MPROC Run Bit

10.4.1.16 CSR 22 "Diag Single Cyc"

EB57 - Verify "Diag Single Cyc"

10.4.1.17 PIA Level Selection

EB60 - PI Level n/0 Interrupts
EB61 - Check PIA Level 0
EB62 - Check Interrupt on Level 1-7
EB63 - CSR 05 "Rqst Interrupt" Clearing

10.4.1.18 IOP Function Words

EB64 - IOP Function 0 - 40+2N Interrupt
EB65 - IOP Function 1 - 40+2N Interrupt
EB70 - IOP Function 4 - Examine
EB71 - IOP Function 5 - Deposit
EB72 - IOP Function 7 - Examine/Increment

10.4.1.19 CSR Bits 25-31

EB73 - CSR 25-26 & 28 KL-side
EB74 - CSR 25-26 & 28 Port-side
EB75 - CSR 27 & 30-31 KL-side
EB76 - CSR 27 & 30-31 Port-side

10.4.1.20 2901 Isolation Tests

AL1 - 2901 0s Test
AL2 - 2901 1s Test

10.4.1.21 2901 Boolean Tests (Direct Input/0)

AL3 - 2901 OR Test - D OR 0
AL4 - 2901 AND Test - D AND 0
AL5 - 2901 NOT_R_AND_S Test - D NOT_RS 0
AL6 - 2901 XOR Test = D XOR 0
AL7 - 2901 XNOR Test - D XNOR 0
AL10 - 2901 R+S Test - D + 0
AL11 - 2901 R-S Test - D - 0
AL12 - 2901 S-R Test - 0 - D

10.4.1.22 2901 Register File Tests

AL13 - 2901 Writing/Reading Registers
AL14 - 2901 Q-Register Test
AL15 - 2901 Register Interference Test

10.4.1.23 2901 ALU Function Tests

AL16 - 2901 OR Test
AL17 - 2901 AND Test
AL20 - 2901 NOT_R_AND_S Test
AL21 - 2901 XOR Test
AL22 - 2901 XNOR Test
AL23 - 2901 R + S Test - No Carry
AL24 - 2901 R + S Test - With Carry
AL25 - 2901 R - S Test - No Carry
AL26 - 2901 R - S Test - With Carry
AL27 - 2901 S - R Test - No Carry
AL30 - 2901 S - R Test - With Carry

10.4.1.24 Register Interference Tests

AL31 - 2901 Q-Register Interference Test
AL32 - 2901 RAM Register Interference Test

10.4.1.25 Q-Register Shifter Tests

AL33 - Q-Register Shift
AL34 - Q-Register Right/Left Shift

10.4.1.26 RAM Register Shifter Tests

AL35 - RAM Register Shift
AL36 - RAM Data Movement Test
AL37 - RAM/Q-Register Data Movement Test
AL40 - RAM Data Movement/Shifting Test
AL41 - RAM/Q-Reg Data Movement/Shifting Test
AL42 - 2901 Register / Constant Mux Test

10.4.2 Hardware Tests - MPROC Module

10.4.2.1 CRAM Tests

MP1 - CRAM Data Test RAR12=0
MP2 - CRAM Data Test RAR12=1
MP3 - CRAM Addressing Test

10.4.2.2 RAR/LAR Tests

MP4 - RAR/LAR Data Path Test - LDCRAM
MP5 - RAR/LAR Data Path Test - RDCRAM

10.4.2.3 Cram Ctrl Register Tests

MP6 - Cram Ctrl Register Test
MP7 - Cram Ctrl Register Test

10.4.2.4 MBUS Error Tests

MP10 - MBUS Error Test (MPROC module only)
MP11 - MBUS Error Test
MP12 - MBUS Error Test (MPROC module only)
MP13 - MBUS Error Test

10.4.2.5 Condition Code Tests

MP14 - CCCbusAvail Test
MP15 - CCGrntCSR Test
MP16 - CCFEQ0 Test
MP17 - CCCSRChng Test
MP20 - CCEbParErr Test
MP21 - CCRcvrBufAFul Test
MP22 - CCRcvrBufBFul Test
MP23 - CCXmtrAttn Test
MP24 - CCEbusRqst Test
MP25 - CCIntrActive Test
MP26 - CCMBSign Test
MP27 - CCMVParChk Test
MP30 - CCCbusParErr Test
MP31 - CCPLiParErr Test
MP32 - CCChanErr Test
MP33 - CCCbLstWd Test

10.4.2.6 Local Storage Tests

MP34 - Basic Local Store Test
MP35 - Local Store Data Test
MP36 - Local Store Address Test
MP37 - Local Store RAM Mode Test

10.4.2.7 Miscellaneous MPROC Tests

MP40 - Jump MUX Test
MP41 - CRAM Ctl Reg Parity Generators Test

10.4.2.8 2910 Basic Instruction Tests

SE1 - JMAP Instruction
SE2 - JZ Instruction
SE3 - JZ Instruction - Effect on Reg/Ctr
SE4 - JZ Instruction - Effect on Reg/Ctr
SE5 - CONT Instruction
SE6 - CJP Instruction
SE7 - CJV Instruction
SE10 - CJS Instruction
SE11 - CJS Instruction
SE12 - CJS Instruction
SE13 - CRTN Instruction
SE14 - CRTN Instruction
SE15 - Stack location #1
SE16 - Stack location #2
SE17 - Stack location #3
SE20 - Stack location #4
SE21 - Stack location #5

SE22 - Increment/Decrement Stack Pointer
SE23 - Increment/Decrement Stack Pointer
SE24 - Increment/Decrement Stack Pointer
SE25 - Increment/Decrement Stack Pointer
SE26 - Increment/Decrement Stack Pointer
SE30 - CJPP Instruction
SE31 - CJPP Instruction
SE32 - CJPP Instruction
SE33 - LDCT Instruction
SE34 - LDCT Instruction
SE35 - Register/Counter
SE36 - JRP Instruction
SE37 - JSRP Instruction
SE40 - JSRP Instruction
SE41 - JSRP Instruction
SE42 - PUSH Instruction
SE43 - PUSH Instruction
SE44 - PUSH Instruction
SE45 - LOOP Instruction
SE46 - LOOP Instruction
SE47 - LOOP Instruction
SE50 - RFCT Instruction
SE51 - RFCT Instruction
SE52 - RFCT Instruction
SE53 - RFCT Instruction
SE54 - RPCT Instruction
SE55 - RPCT Instruction
SE56 - TWB Instruction
SE57 - TWB Instruction
SE60 - TWB Instruction
SE61 - TWB Instruction
SE62 - TWB Instruction
SE63 - TWB Instruction
SE64 - Stack Interference Tests
SE65 - Stack Interference Tests
SE66 - Reg/Ctr Interference Tests
SE67 - Reg/Ctr Interference Tests
SE70 - Full Speed Sequencer Test

10.4.3 Hardware Tests - CBUS Module

10.4.3.1 Basic CBUS Tests

CB1 - Fmtr Data Loopback
CB2 - PLI Buffer Data Loopback

10.4.3.2 CBUS Data Transfer Tests

CB3 - CBUS to EBUF Data Transfer
CB4 - 2901 to CBUS Data Transfer
CB5 - CBUS to 2901 Multiple Word Transfer
CB6 - 2901 to CBUS Multiple Word Transfer

CB7 - CBUS to EBUS Data Transfer
CB10 - EBUS to CBUS Data Transfer

10.4.3.3 Mover/Formatter Tests

CB11 - Fmtr Cleared by "Port Clear"
CB12 - Fmtr Basic Up Shifting Test
CB13 - Fmtr Up Shift 4 Test - BUS Ctl=Cbus
CB14 - Fmtr Up Shift 4 Test - BUS Ctl=Fmtr
CB15 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=0
CB16 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=0
CB17 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=1
CB20 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=1
CB21 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=0
CB22 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=0
CB23 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=1
CB24 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=1
CB25 - Fmtr Basic Down Shifting Test
CB26 - Fmtr Down Shift 4 Test
CB27 - Fmtr Down Shift 4 (from PLIN) Test - SELRHTNIB=0
CB30 - Fmtr Down Shift 4 (from PLIN) Test - SELRHTNIB=0
CB31 - Fmtr Down Shift 4 (from PLIN) Test - SELRHTNIB=1
CB32 - Fmtr Down Shift 4 (from PLIN) Test - SELRHTNIB=1
CB33 - Fmtr Down Shift 8 (from PLIN) Test - SELRHTNIB=0
CB34 - Fmtr Down Shift 8 (from PLIN) Test - SELRHTNIB=0
CB35 - Fmtr Down Shift 8 (from PLIN) Test - SELRHTNIB=1
CB36 - Fmtr Down Shift 4 (from PLIN) Test - SELRHTNIB=1
CB37 - Fmtr Ring Buffer Test

10.4.3.4 Miscellaneous CBUS Tests

CB40 - CMUX Selected by PLINTOCMUX
CB41 - CBUF Loaded Correctly
CB42 - PMUX Selected by MPCBUFTOPLOUT
CB43 - PMUX Selected by MPCBUFTOPLOUT/MPZEROLFTNIB
CB44 - DMUX Selected Correctly
CB45 - CBUS Parity Checking - CBUS to CBUS Module
CB46 - CBUS Parity Checking - CBUS Module to CBUS
CB47 - CBUS Parity Checking - CBUS to CBUS Module
CB50 - T Field Timing
CB51 - CBUS Store
CB52 - CBUS Input Buffer

10.4.3.5 Parity Predictor Tests

CB53 - Parity Predictor - CBIN to Fmtr/MPINDSTCOMP=0
CB54 - Parity Predictor - CBIN to Fmtr/MPINDSTCOMP=1
CB55 - Parity Predictor - Fmtr to PLOUT
CB56 - Parity Predictor - PLIN to Fmtr/MPCOREDUMP=0
CB57 - Parity Predictor - PLIN to Fmtr/MPCOREDUMP=1

CB60 - Parity Predictor - Fmtr to CBOUT

10.5 Debug Test

10.5.1 Basic Description

This test allows the selection of different operations that may be done on port. In some cases, probably rarely, a field engineer may wish to diagnose an port by doing particular operations. More often this type of testing may be useful when repairing individual modules if the technician doing the repair is very familiar with the module and how it works. This test provides the ability to perform a basic operation such as writing a register or doing something with microcode and then inspecting the results to see what happened.

Normally, the port tests provide sufficient detection and isolation of faults to fix the port. Occasionally, a person trying to fix the port may encounter an unusual problem not detected by the diagnostic or having strange symptoms. This test may allow some understanding of the problem to be gained.

This test can also be used to debug microcode using the breakpoint facilities. And it can be used to run small microcode routines of the user's own devising.

10.5.2 Commands Available

The commands available in debug mode are as follows, which are typed in response to the prompt 'DEBUG> ':

```
NAME          DESCRIPTION
----          -----------

General Commands:
-----------------

HELP          Print this message
EXIT          Exit DEBUG mode
DDT           Enter DDT (Type RTN$X to return)

SET opt arg     Set/clear/print options
TAKE file ext   Take commands from a file
NTAKE file ext  Take commands from file (unechoed)

Device Selection Commands:
--------------------------

CONFIG        Determine/print configuration
CONPNT        Print configuration
SELECT        Select device to test

Starting, Stopping, Initializing the Port:
------------------------------------------

RESET         Issue EBUS Reset
CLEAR         Issue a "Port Clear"
```

START adr	Start the port micro-sequencer
STOP	Stop the port micro-sequencer
CONT	Continue the port micro-sequencer
SSTEP cnt	Single step the port micro-sequencer
STRACE cnt	Single step and trace execution
SPRINT cnt	Print single step history data
SCLEAR	Clear single step history data
SSINIT	Set up single step addr/data/history

Handling Port Registers:

ECSR	Examine CSR Register
DCSR dat	Deposit CSR Register
ZCSR	Zero CSR Register
EEBUF	Examine EBUF register
DEBUF dat	Deposit EBUF register
ZEBUF	Zero EBUF register
ELAR	Examine LAR register
DRAR dat	Deposit RAR register
ZRAR	Zero RAR register

Handling Port CRAM Locations:

The next few commands must have at least one argument and possibly two. Two arguments give the range of CRAM address to be altered or inspected. If only one argument is given the range is only one location. If the deposit command is given, the diagnostic will ask what to deposit into each location.

DCRAM adr,adr	Deposit into CRAM
BCRAM adr,adr	Deposit into CRAM by fields
ACRAM adr,adr	Alter CRAM
ECRAM adr,adr	Examine CRAM
LCRAM adr,adr	List CRAM
ZCRAM adr,adr	Zero CRAM

2901 Commands:

E2901 reg,reg	Examine 2901 registers
D2901 reg,dat	Deposit 2901 register
Z2901 reg,reg	Zero 2901 registers

Local Storage Commands:

ELOCS adr,adr	Examine local storage
DLOCS adr,dat	Deposit local storage
ZLOCS adr,adr	Zero local storage

Special Scope Functions:

```
     -----------------------

     LCONI       Loop on CONI
     LCONO dat   Loop on CONO
     LDATAI      Loop on DATAI
     LDATAO dat  Loop on DATAO
     LROUTN dat  Loop on user supplied routine at DROUTN:

     Mark Bit Commands:
     -----------------

     MARK adr,adr   Set mark bit
     RMARK adr,adr  Remove mark bit
     CMARK adr,adr  Clear all mark bits
     LMARK adr,adr  List mark bits

     Breakpoint Commands:
     -------------------

     BREAK adr,adr   Set breakpoint
     RBREAK adr,adr  Remove breakpoint
     CBREAK adr,adr  Clear all breakpoints
     LBREAK adr,adr  List breakpoints
     PBREAK adr      Proceed from breakpoint
     WBREAK          Print breakpoint data

     Microcode Commands:
     ------------------

     LOAD tst    Load microcode from Test "tst" into CRAM
     LIST tst    List microcode from Test "tst"
     FLOAD       Load microcode from file into CRAM
     FVERFY      Verify microcode loaded from file
     FLIST       List microcode in file
     FEXAM       Examine microcode in file

     Miscellaneous Commands:
     ----------------------

     TRANSL dat  Translate data to CSR bits
     CCODE       Print condition codes
     ESTACK      Examine 2910 stack contents
     FILLNX dat  Fill CRAM with "J=.+1,CTL=dat"
     FILLPC dat  Fill CRAM with "J=.,CTL=dat"
     ENEXT       Examine next
     DNEXT arg   Deposit next

     IPRINT      Print accumulated interrupt activity
     ICLEAR      Clear accumulated interrupt activity
     IINIT       Init interrupt system

     Data Transfer Commands:
     ----------------------

     RDINIT    Initialize everything for a read
     WRINIT    Initialize everything for a write
```

BPRINT n	Print buffer contents (n locations)
CCWPNT	Print CCW list
LOGPNT	Print logout data
COMPAR	Compare data buffer

Switches:

SWITCH	Enter switch information
SWPRIN	Print current switch settings
NOPNT	Complement "no print" switch
PNTLPT	Complement "print lpt" switch
DING	Complement "bell on error" switch
LOOPER	Complement "loop on error" switch
ERSTOP	Complement "halt on error" switch
PALERS	Complement "print all errors" switch
RELIAB	Complement "reliability mode" switch
TXTINH	Complement "text inhibit" switch
OPRSEL	Complement "operator select" switch
TRACE	Complement "trace test" switch
INHFLT	Complement "inhibit fault isolation" switch
INHMSG	Complement "inhibit error message" switch
LOOPGM	Complement "loop on pgm" switch
LOOPTS	Complement "loop on test" switch
RUNALL	Complement "run all test segments" switch
DSPEAR	Complement "disable SPEAR reporting" switch
UDEBUG	Complement "user mode debug" switch
MDEBUG	Complement "error message debug" switch
SDEBUG	Complement "test debug" switch
LDEBUG	Complement "load microcode debug" switch
IDEBUG	Complement "fault isolation debug" switch
NIPORT	Complement "Select NI port" switch
CIPORT	Complement "Select CI port" switch
MPROC	Complement "missing MPROC module" switch
MCBUS	Complement "missing CBUS module" switch

10.5.3 Detailed Command Description

In more detail these commands are as follows:

10.5.3.1 General Commands

HELP

This command prints a one line description similar to the above for each command.

EXIT

This command returns control to normal test dispatching at the 'What Test -' question.

DDT

This command invokes DDT. When RTN$X (POPJ 17,0) is executed, the diagnostic resets the PI system in exec mode and returns to 'DEBUG>' question. DDT can also be entered by typing Control-D and when done typing RTN$X. When invoked in this fashion, the diagnostic does not reset the PI system.

SET

This command is used to set, clear, or print program options. The option is specified as the first argument and the option setting as the second argument. The options that may be dealt with are as follows:

- SET ADDR adr.

  This command sets the start address of the microsequencer. Whenever a START command or a SSINIT command is given, the default address given here is used. A START command may also explicitly specify the starting address. No commands other than SET ADDR alter the default starting address.

  The initial default address is location 0. To find out the current default address, type SET ADDR <CR>.

- SET CSR data.

  This command sets the initial CSR register starting contents. When a subsequent START or SSINIT command is given, this data is written to the CSR when starting the microsequencer. This allows the state of CSR bits to be set up - such as interrupt enable and PI level bits.

  The initial value is 0. When a START or single step command is given with the default CSR data equal to 0, the diagnostic writes 14 to the CSR register (MPRUN, PI Level 4) or 20014 if single stepping (SINCYC bit included).

  To find out the current start data, type SET CSR <CR>.

- SET PARITY arg.

  This command affects the automatic parity generation when loading microcode into the CRAM. If the flag is "YES" the diagnostic computes correct parity before it loads any microcode. If "NO" the diagnostic assumes the parity has already computed and does not alter the parity bit specified in the microcode data.

This flag only affects microcode explicitly loaded in the DEBUG test. This flag does not affect loading microcode from a file. When operational microcode is loaded in this fashion, parity is not affected.

The initial value of the flag is "YES". To find out the current setting, type SET PARITY <CR>.

o SET EBUF arg.

This command affects the automatic preservation of the EBUF when single stepping or stopping the microsequencer. If "preserve EBUF" flag is "YES" the diagnostic will read the EBUF before reading the LAR and then write back the data afterwards. If the flag is "NO" the contents of the EBUF will be destroyed every time the RAR is written or the LAR read.

The initial value of the flag is "YES". To find out the current setting, type SET EBUF <CR>.

o SET LENGTH len.

This command sets the buffer length to be used when generating CCW lists. Any length up to 102400 (decimal) words may be specified. However, note that the buffer is only 512 words long and each CCW in the CCW list points to the same starting location. This is so that a 512 word buffer can be used and much longer data transfers can be specified.

The initial length is 128 words. To find out the current setting, type SET LEN <CR>.

o SET PAT n.

This command specifies the data pattern to be used by commands which generate CCW lists and set up buffer contents and do buffer compares. Any data pattern from 1 to 126 (octal) may be entered.

The initial value is 1 (data pattern zeros). To find out the current setting, type SET PAT <CR>.

o SET PATLIS arg,arg.

This command prints what data patterns are available. One or two arguments may be given to specify a range of data patterns. The diagnostic prints a description for each data pattern in the range specified. If no arguments are specified, all data patterns will be printed.

o SET FNAME name ext

The diagnostic allows a microcode file to be loaded into CRAM or listed on the terminal. The commands used do not allow a microcode file name to be specified. The name of the file used can be changed by this command.

The initial file name is "CI20.ULD". To find out the current setting, type SET FNAME <CR>.

- SET WHAT

  This command prints the current selection of each option.

- SET HELP

  This command prints a help message describing each option available to the set command.

TAKE file ext

This command takes commands from a file. The file may be specified as an argument to the command. If no file name is specified, the default file name is used. The default file name is either "DFPTA.CMD" or the last file specified.

Any legal command may be specified including another TAKE command. The presence of a TAKE or DDT or EXIT (from DFPTA) command in the take file will terminate use of the current file. The diagnostic prints each command and result on the terminal as it executes each command.

The file specified must be on the load device known to MAGMON or D20MON or DIAMON. If not, an error message is printed and the TAKE command is aborted.

If any error occurs while processing the take file, the TAKE command is aborted. Errors include any unexpected errors such as unrecognized command, failures to start the port, argument errors, etc.

NTAKE file ext

This command is identical to the TAKE command with the single exception that the commands executed are not echoed on the terminal. All output resulting from execution of the commands is echoed.

10.5.3.2 Device Selection Commands

CONFIG

This command causes the diagnostic to reconfigure. It determines whether or not an NI or CI port exists, and if each is selectable by the diagnostic. Then the configuration is printed and the port(s) to test are queried as follows:

```
| Port Configuration:
|   Port  Exists  Selectable  Selected
|    NI    Yes       Yes         Yes
|    CI    Yes       Yes         Yes
```

```
Test NI port? (Y or N) - N

Test CI port? (Y or N) - Y

[CI port selected]
```

CONPNT

This command prints the configuration determined at last configuration as modified by any selection of ports to test. For example:

```
Port Configuration:
  Port  Exists  Selectable  Selected
   NI    Yes       Yes         No
   CI    Yes       Yes         Yes
```

SELECT port,port

This command allows the current selection of ports to be modified. There are two alternate formats for this command. First, SELECT <CR> will cause the diagnostic to query which port to select as follows:

```
What test - SELECT

Test NI port? (Y or N) - N

Test CI port? (Y or N) - Y

[CI port selected]
```

The selection may be stated in the SELECT command by typing SELECT NI <CR> or SELECT CI <CR> or SELECT NI,CI <CR>. An example follows:

```
What test - SELECT NI,CI

[NI port selected]
[CI port selected]
```

10.5.3.3 Starting, Stopping, Initializing the Port

The START, SSTEP or STRACE commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

RESET

Issue EBUS Reset. This does a COND APR,200000. In user mode a "Port Clear" is done instead.

CLEAR

Issue a "Port Clear". A 400000 is written to the CSR register.

START adr

The port is started at address specified. The procedure is as follows:

- o Determine start address. If no address is specified, the start address used is the default address given by the SET ADDR command.
- o Determine start CSR data. This is the data specified by a SET CSR command. Insert the necessary bit "MPRUN" and ensure "SINCYC" bit is not set.
- o Read the CSR register, if the port is already running or error bits are set, stop the port or clear error bits as necessary.
- o Write the start address to the RAR
- o Write CSR data to the CSR

CONT

This command is identical in function to the START command with the exception of the start address. In addition, if the port is not already stopped, an error message is printed and the start sequence aborted. The start address is the last "next" address obtained from a single step command.

The only purpose of this command is to start up the port after single stepping it for a while.

STOP

Stop the port micro-sequencer and print out the contents of the LAR. If the CSR is inaccessible, or if error bits CRAM PE or MBUS Error are set, an error message is printed.

SSTEP n and STRACE n

Single step the port micro-sequencer "n" times. Before this command is done the first time, the start data and start CSR data must be set up using the SSINIT command. Thereafter, the address written to the RAR each time the port is single stepped is the "last address" read from the LAR after the prior single step.

STRACE causes the single step history data to be printed at the same time that single steps are being done. The data is printed in the same format as the SPRINT command except for the order in which the single steps are printed.

The port is stopped or error bits are cleared from the CSR register as necessary before any single steps are done.

If CRAM PE or MBUS Error bits become set in the CSR register, a message is printed and the single stepping is aborted.

To abort the single step procedure at any time, type altmode.

SPRINT n

Print single step history data. Data for the last "n" (maximum 128) single steps are printed in the following format. If no argument is given, no more than 20 single steps are printed.

```
DEBUG> SPRINT 4

 RAR   LAR   CSR (final)   SS#
0000  0100  000000 020010  12
0100  0230  000000 020010  11
0230  0231  000000 020010  10
0231  0232  000000 020010  7
```

SCLEAR

Clear single step history data.

SSINIT

Set up single step data. This sets the single step address to be the default address specified by the SET ADDR command. The single step CSR data is set to the default data specified by the SET CSR command. Also, the single step history data is cleared.

This command MUST be given in order to execute the first single step. If not given the port will start at the last address executed by a single step.

10.5.3.4 Handling Port Registers

Each of the EBUF commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

ECSR

Examine CSR Register. The contents are printed in octal and translated into English as well.

DCSR dat

Deposit CSR Register. The data expected is up to 6 octal digits.

ZCSR

Zero CSR Register. This also stops the port if it is running.

EEBUF

Examine EBUF register. The data is printed in halfword format.

The port is stopped or error bits are cleared from the CSR register as necessary before reading the EBUF. Then the "Test EBUF" bit is set in the CSR register if it is not already set and the EBUF is read.

DEBUF dat

Deposit EBUF register. The data expected is up to 12 octal digits.

The port is stopped or error bits are cleared from the CSR register as necessary before reading the EBUF. Then the "Test EBUF" bit is set in the CSR register if it is not already set and the EBUF is written.

ZEBUF

Zero EBUF register

The port is stopped or error bits are cleared from the CSR register as necessary before reading the EBUF. Then the "Test EBUF" bit is set in the CSR register if it is not already set and the EBUF is zeroed.

ELAR

Examine LAR register and print it as "LAR/ 0000". This is actually a 13 bit register. The right most bit is the bit written to the RAR register to specify which half of the CRAM to access. This bit is not printed.

DRAR dat

Deposit RAR register. The data consists of a 12 bit CRAM address followed by one bit specifying left or right half. Example:

```
¦ DEBUG> DRAR 0           ; Write location 0 (right half)
¦ DEBUG> DRAR 1           ; Write location 0 (left half)
¦ DEBUG> DRAR 10          ; Write location 4 (right half)
¦ DEBUG> DRAR 11          ; Write location 4 (left half)
```

ZRAR

Zero RAR register.

10.5.3.5 Handling Port CRAM Locations

Each of the following commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect. The diagnostic also ensures that "Test EBUF" bit is not set and "Sel LAR" bit is not set.

Commands that read or write CRAM locations are given up to 2 arguments. The action in each case is as follows:

- No arguments - The CRAM addresses used are those last used. So if a zero CRAM command is given to clear locations 1001 through 1004, a subsequent examine cram command with no arguments would examine locations 1001 through 1004.
- One argument - The range of CRAM locations is taken as addr1,addr1.
- Two arguments - The range of CRAM locations is taken as addr1,addr2.

DCRAM adr,adr

Deposit into CRAM locations specified either by explicit arguments or by default. The format of the data is the same format produced as output of the MICRO2 assembler. Example:

```
DEBUG> DCRAM 10

Type xxxx,xxxx,xxxx,xxxx,xxxx
0010 ::   11,22,33,44,55
```

BCRAM adr,adr

Deposit into CRAM locations specified either by explicit arguments or by default. The data is asked for each location, one field at a time. The initial data to be written to each CRAM location is zero. Example:

```
DEBUG> BCRAM 10

Type  value,value... or ^,CR,^Z,field name
0010 ::
J -
PAR -               Type:  value<CR> value<CR> ...
```

```
|  OENA -                 or:  value,value,...<CR>
|  MGC -                  or:  ^ to back up to the previous
|  SORC -                          value
|  FUNC -                 or:  <CR> to leave a value unchanged
|  DEST -                 or:  field name such as MGC or FUN or
|  CENA -                          A or M (for MGC) .. to skip to
|  RAM -                           a particular field
|  A -                    or:  ^Z to terminate input and store
|  B -                             the CRAM location
|  SK -                   or:  $ (altmode) to abort the command
|  BUS -                           and not store anything
|  CRY -
|  CTL -
|  TIME -
|  SPARE -
|  MARK -
```

Note: Everything can be typed on one line, such as this example to change A, B, and MGC fields only:

```
|        DEBUG> BCRAM 10
|
|        Type  value,value... or ^,CR,^Z,field name
|        0010 ::
|        J - A,3,4,M,1234,^Z
|
|        DEBUG>
```

ACRAM adr,adr

Alter CRAM locations specified either by explicit arguments or by default. The data is asked for each location, one field at a time. Example:

```
|  DEBUG> ACRAM 10
|
|  Type  value,value... or ^,CR,^Z,field name
|  0010 ::
|  0 J -
|  0 PAR -                 Type:  value<CR> value<CR> ...
|  0 OENA -                  or:  value,value,...<CR>
|  33 MGC -                  or:  ^ to back up to the previous
|  0 SORC -                           value
|  1 FUNC -                  or:  <CR> to leave a value unchanged
|  0 DEST -                  or:  field name such as MGC or FUN or
|  0 CENA -                           A or M (for MGC) .. to skip to
|  0 RAM -                            a particular field
|  0 A -                     or:  ^Z to terminate input and store
|  0 B -                              the CRAM location
|  22 SK -                   or:  $ (altmode) to abort the command
|  0 BUS -                            and not store anything
|  1 CRY -
|  0 CTL -
|  0 TIME -
|  0 SPARE -
```

```
:  0 MARK -
```

Note: Everything can be typed on one line, as in the example under "BCRAM" command above.

ECRAM adr,adr

Examine CRAM locations specified either by explicit arguments or by default. The data is printed in MICRO2 assembler format. Example:

```
!  DEBUG> ECRAM 10,12
!
!  0010 / 0000 0001 0400 0000 0400
!  0011 / 0000 0001 1400 0000 0440
!  0012 / 0000 0001 2400 0000 0500
```

LCRAM adr,adr

List CRAM locations specified either by explicit arguments or by default. The data is printed in bit field format. Example:

```
!  DEBUG> LCRAM 0,3
!
!  ADDR/ JUMP PAR OUT MGC SOR FUN DES CC RAM PA PB SK BUS CRY CTL  TM SP MK
!     0/    0  0   0    0  0   1   0   0  0   0  0  0  0   0  JZ    0  0  0
!     1/    0  0   0    0  0   3   0   0  0   0  0  0  0   0  JZ    0  1  0
!     2/    C  0   0    0  0   5   0   0  0   0  0  0  0   0  JZ    0  2  0
!     3/    0  0   0    0  0   7   0   0  0   0  0  0  0   0  JZ    0  3  0
```

ZCRAM adr,adr

Zero CRAM locations specified either by explicit arguments or by default.

10.5.3.6 2901 Commands

Each of the following commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

These commands may be given up to 2 arguments. The action in each case is as follows:

- o No arguments - The 2901 registers used are those last used. So if registers 0 through 4 are examined, a subsequent examine command with no arguments would examine registers 0 through 4.

- o One argument - The range of 2901 registers is taken as reg1,reg1.
- o Two arguments - The range of 2901 registers is taken as reg1,reg2

E2901 reg,reg

Examine the contents of the range of 2901 registers specified. Microcode is loaded into the port to accomplish this, but no 2901 registers or local storage locations are affected. And the original contents of the CRAM location used are restored.

D2901 reg,data

Deposit the data into the 2901 register specified. Microcode is loaded into the port to accomplish this, and only the 2901 register specified is affected. The original contents of the CRAM location used are restored.

To deposit data into successive registers, additional data can be typed. For example, to load registers 3,4,5,6 with 333,444,555,666 respectively, type the following:

```
DEBUG> D2901 3,333,444,555,666
```

Z2901 reg,reg

Zero the range of 2901 registers specified. Microcode is loaded into the port to accomplish this. The original contents of the CRAM location used are restored.

10.5.3.7 Local Storage Commands

Each of the following commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

These commands may be given up to 2 arguments. The action in each case is as follows:

- o No arguments - The local storage addresses used are those last used. So if a zero local storage command is given to clear locations 101 through 104, a subsequent examine local storage command with no arguments would examine locations 1001 through 1004.

- o One argument - The range of local storage locations is taken as addr1,addr1.
- o Two arguments - The range of local storage locations is taken as addr1,addr2

ELOCS adr,adr

Examine local storage addresses specified by the range of addresses. Microcode is loaded into the port to accomplish this and several 2901 registers are destroyed in the process. No local storage locations are affected. And the original contents of the CRAM locations used are restored.

DLOCS adr,dat

Deposit local storage address "adr" with 36 bit octal data "dat". Microcode is loaded into the port to accomplish this and several 2901 registers are destroyed in the process. No local storage locations are affected. And the original contents of the CRAM locations used are restored.

To deposit data into successive locations, additional data can be typed. For example, to load locations 103,104,105,106 with 333,444,555,666 respectively, type the following:

```
DEBUG> DLOCS 103,333,444,555,666
```

ZLOCS adr,adr

Zero local storage addresses specified by the range of addresses. Microcode is loaded into the port to accomplish this, but no 2901 registers or local storage locations are affected. And the original contents of the CRAM locations used are restored.

10.5.3.8 Special Scope Functions

These commands allow scope loops to be set up for only doing a CONI, CONO, DATAI, or DATAO. The diagnostic will repeatedly perform the operation and not print out anything until altmode is typed.

LCONI

Loop on CONI instruction (CONI 564,1 or CONI 574,1)

LCONO dat

Loop on CONO instruction (CONO 564,dat or CONO 574,dat). This writes the CSR register continuously. The data to write to CSR must be specified up to 6 octal digits.

LDATAI

Loop on DATAI instruction (DATAI 564,1 or DATAI 564,1). This may read the LAR or a CRAM location or the EBUF depending on the state of CSR bits "Test EBUF" and "Select LAR". To set up the CSR as desired just do an EEBUF or ELAR or ECRAM instruction just before this command.

LDATAO dat

Loop on DATAO instruction (DATAO 564,1 or DATAO 574,1). The data to write must be specified. This may write the LAR or a CRAM location or the EBUF depending on the state of CSR bit "Test EBUF" and the leftmost bit of the data given to write. To set up the CSR as desired just do an EEBUF or ELAR instruction just before this command.

LROUTN dat

This command allows the user to enter a routine at label DROUTN: and loop on it continuously until an altmode is typed. There are 15 locations free at DROUTN: to insert code. An octal argument is optional and is available in location "ARGUM" if typed.

10.5.3.9 Mark Bit Commands

Each CRAM word has a mark bit that can be set or cleared. It is used only to sync a scope on. Parity is not calculated for this location.

The diagnostic does not keep track of which locations have mark bits set. If it is requested to list marked CRAM locations, the diagnostic just looks for locations with the mark bit set.

Each of these commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

MARK adr1,adr2,adr3,...

Set mark bit at locations adr1,adr2,adr3,...

RMARK adr1,adr2,adr3,...

Remove mark bit from locations adr1,adr2,adr3,...

CMARK adr1,adr2

Clear all mark bits between adr1 and adr2. If only one argument is given only that one location is rewritten without the mark bit set. If no arguments are supplied, all mark bits are cleared throughout the CRAM.

LMARK adr1,adr2

List all locations with mark bit set between adr1 and adr2. If only one argument is given only that one location is checked. If no arguments are supplied, all location with the bit set are listed throughout the CRAM.

10.5.3.10 Breakpoint Commands

A breakpoint consists of a location with bad parity. The diagnostic does not keep track of these locations itself. If it is requested to list breakpoints, it just looks for locations with bad parity.

Each of these commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

BREAK adr1,adr2,adr3,...

Set breakpoints at locations adr1,adr2,adr3,...

RBREAK adr1,adr2,adr3,...

Remove breakpoints from locations adr1,adr2,adr3,...

CBREAK adr1,adr2

Clear all breakpoints between adr1 and adr2. If only one argument is given only that one location is rewritten with good parity. If no arguments are supplied, all breakpoints are removed throughout the CRAM.

LBREAK adr1,adr2

List breakpoints between adr1 and adr2. If only one argument is given only that one location is checked for bad parity. If no arguments are supplied, all breakpoints are listed throughout the CRAM.

PBREAK adr

Proceed from breakpoint. The address given will be used as the restart address. If none is given, the address read by the WBREAK command is used. This command causes several things to be done:

1. The CRAM location to restart at is read and rewritten with good parity.
2. The RAR is written with the start address.

3. The EBUF is written with the EBUF data read by the WBREAK command.

4. The port is restarted.

WBREAK

Assuming the port has just halted because of a CRAM PE, this command is used to print breakpoint related data - the address stopped at and the contents of the EBUF.

This command also saves this data so that a subsequent PBREAK command can be used to restart the port at the correct place, restoring the EBUF also. Note that restarting is not possible unless the instruction to restart at is an unconditional jump instruction.

10.5.3.11 Microcode Commands

Each of these commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

Test names are given as "Test Class" "Test Number", where test class is:

- SE or S - 2910 sequencer tests
- MP or M - Mproc module tests
- AL or A - 2901 ALU tests
- CB or C - CBus module tests
- EB or E - EBus module tests

Sample test names are SE1, SE27, EB1, E1, AL20, ...

LOAD tst

Load microcode from Test "tst" into CRAM and verify the result. Example:

```
DEBUG> LOAD SE27

[Done - Number of CRAM locations loaded/verified = 17.]
```

LIST tst

List microcode from Test "tst". All locations are listed in the order that they would be loaded, not necessarily in order by CRAM address. Example:

```
DEBUG> LIST SE30

ADDR/ JUMP PAR OUT MGC SOR FUN DES CC RAM PA PB SK BUS CRY CTL  TM SP MK
   0/    2  0   0    0  0   0   0   0  0   0  0  1  0    0 CJPP  0  0  0
   1/   10  1   0    0  0   0   0   0  0   0  0  0  0    0 JMAP  0  0  0
   2/    4  1   0    0  0   0   0   0  0   0  0  3  0    0 CJPP  0  0  0
   3/   11  0   0    0  0   0   0   0  0   0  0  0  0    0 JMAP  0  0  0
   ...
```

FLOAD

Load microcode from file into CRAM. The file used is the default file name CI20.ULD or the file last specified by the SET FNAME command. DIAMON or D20MON or MAGMON must be loaded in order to load the microcode from the selected load device. The diagnostic prints an error message if it cannot access the microcode file. This command does not verify the microcode it has loaded. Example:

```
DEBUG> FLOAD

[Reading file: CI20.ULD]

[Number of CRAM locations loaded = 3040.]

[Ucode version 676]
```

FVERFY

This command causes the microcode previously loaded into CRAM from a file to be verified. Up to 4 verify errors are printed. Example:

```
DEBUG> FVERFY

[Reading ucode file: CI20.ULD]

Verify error at CRAM location 0000
 Correct: 0001 0000 1631 6702 0060
 Actual:  0000 0000 0400 0000 0000

Verify error at CRAM location 0001
 Correct: 0025 6000 4315 6542 0060
 Actual:  0000 0000 1400 0000 0040

Verify error at CRAM location 0007
 Correct: 0004 4000 7510 7542 0060
 Actual:  0000 0000 7400 0000 0340
```

```
Verify error at CRAM location 0025
 Correct: 0031 4145 4731 6723 0060
 Actual:  0000 0002 5400 0000 1240

[Number of CRAM locations verified = 3040.  Errors = 27.]
```

FLIST

List microcode in file in field format. The file used is the default file name CI20.ULD or the file last specified by the SET FNAME command. DIAMON or D20MON or MAGMON must be loaded in order to obtain the microcode from the selected load device. The diagnostic prints an error message if it cannot access the microcode file. Example:

```
DEBUG> FLIST

[Reading ucode file: CI20.ULD]

ADDR/ JUMP PAR OUT MGC SOR FUN DES CC RAM PA PB SK BUS CRY CTL  TM SP MK
   0/    2  0   0    0  1   6   3   0  0  16 16  2  0    0  CJP  0  0  0
   1/    2  1   0    0  7   5   1   0  0   0  0  2  0    0  CJP  0  0  0
   2/    7  0   1    0  4   3   1   1  0  16  0  2  0    0  CJP  0  0  0
   ...
```

FEXAM

List microcode in file in MICRO2 format. The file used is the default file name CI20.ULD or the file last specified by the SET FNAME command. DIAMON or D20MON or MAGMON must be loaded in order to obtain the microcode from the selected load device. The diagnostic prints an error message if it cannot access the microcode file. Example:

```
DEBUG> FEXAM

 [Reading ucode file: CI20.ULD]

 0000 / 0001 0000 1631 6702 0060
 0001 / 0025 6000 4315 6542 0060
 0007 / 0004 4000 7510 7542 0060
 ...
```

10.5.3.12 Miscellaneous Commands

TRANSL dat

This is used if a description of the CSR register is not at hand. The data (36 bits) is translated into English. Example:

```
DEBUG> TRANSL 3000
```

```
CSR: 000000 003000 FQErr  MVErr
```

FILLNX dat

This command fills all of CRAM with special data "J=.+1,CTL=dat". The argument specified is the control field for the microsequencer. This may be useful when testing out dispatching of the microsequencer.

If the port is running or halted with an error condition, the diagnostic stops the port and clears any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

Example: Fill CRAM with JMAP (unconditional jump) instructions:

```
DEBUG> FILLNX 2

[Number of CRAM locations loaded = 4096.]

Cram now contains:

0000/ JMAP J=1
0001/ JMAP J=2
...
7776/ JMAP J=7777
7777/ JMAP J=0
```

FILLPC dat

This command fills all of CRAM with special data "J=.,CTL=dat". The argument specified is the control field for the microsequencer. This may be useful when testing out dispatching of the microsequencer.

If the port is running or halted with an error condition, the diagnostic stops the port and clears any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

Example: Fill CRAM with JMAP (unconditional jump) instructions:

```
DEBUG> FILLPC 2

[Number of CRAM locations loaded = 4096.]

Cram now contains:

0000/ JMAP J=0
0001/ JMAP J=1
...
7776/ JMAP J=7776
7777/ JMAP J=7777
```

CCODE

This command determines the setting of each condition code and prints it out. It loads special microcode to do this but restores the original contents afterwards.

Some condition codes cannot be seen in their proper state since this command may change them. These are "CSR Changed", "MB Sign", and "F Eql Zero", but they are printed anyway. For example:

```
What test - CCODE

CC On:   CBAV FEQO CSRC MVRP PLPE CBLW
CC Off:  GCSR EBPE AFUL BFUL XATN ERQS IACT MSGN CBPE CHER
```

ESTACK

This command determines the contents of the internal 2910 microsequencer stack. It loads special microcode to do this but restores the original contents afterwards. For example:

```
What test - ESTACK

2910 Stack:
 Location   Contents (addr)
    Top         3042
    -1           733
    -2           140
    -3          5732
    -4            10
```

ENEXT

ENEXT does an "examine next" function. Whatever examine or deposit operation was previously done, the next location or register is examined. The commands which set up for an examine next are: ECRAM, DCRAM, BCRAM, ACRAM, ZCRAM, LCRAM, E2901, D2901, Z2901, ELOCS, DLOCS, and ZLOCS.

DNEXT arg

DNEXT does an "deposit next" function. Whatever examine or deposit operation was previously done, the data given is deposited into the next location or register. No argument is expected on the BCRAM or ACRAM type of deposit, the data for each field of the microword will be queried. The commands which set up for an examine next are: ECRAM, DCRAM, BCRAM, ACRAM, ZCRAM, LCRAM, E2901, D2901, Z2901, ELOCS, DLOCS, and ZLOCS.

IPRINT

This command prints accumulated interrupt activity. The data for up to 20 interrupts are held in an interrupt data table. This data can be printed with this command.

Printing can be aborted by typing an altmode or suppressed by typing a Control-O.

ICLEAR

This command clears the accumulated interrupt activity. Subsequent interrupts are numbered 1,2,...

IINIT

In the event the PI system got shut off in exec mode, this command may be given to initialize the interrupt system and set up 40+2n addresses. This may be useful in the event the diagnostic shuts off the PI system because of too many interrupts - this occurs after 8192 interrupts.

10.5.3.13 Data Transfer Commands

RDINIT

This command performs initialization in preparation for a read (port writing into KL10 memory). The diagnostic builds a CCW list that will transfer the number of words specified by the last SET LEN command. It also initializes the buffer by filling it with the data pattern "400000,,000001".

WRINIT

This command performs initialization in preparation for a write (port reading KL10 memory). The diagnostic builds a CCW list that will transfer the number of words specified by the last SET LEN command. It also initializes the buffer by filling it with the data pattern selected by the last SET PAT command.

BPRINT n

This command prints buffer contents (n locations). If no length is specified the diagnostic prints the entire buffer the length of which is determined by a previous SET LEN command. For example:

```
| DEBUG> BPRINT 7
|
| Wrd 0.  777777 777777   777777 777777   777777 777777   777777 777777
| Wrd 4.  777777 777777   777777 777777   777777 777777
```

CCWPNT

This command prints the CCW list last generated by a RDINIT, WRINIT, or START command. Example:

```
DEBUG> CCWPNT

CCW List:
  #  Op Code     Word Cnt  Buffer Adr     This data transfer is 100000
  -  -------     --------  ----------     words long.  The buffer is
  1  Dt-fwd         512    00047000       only 512 words long so the
 ..                                       buffer address remains the
304  Dt-fwd-hlt     160    00047000       same for the entire transfer.
```

LOGPNT

This command prints the logout data last stored by the channel upon receipt of the "store" command issued by the microprocessor. Example:

```
DEBUG> LOGPNT

Channel-7 Logout Data
Loc    Contents
 1     ICW: 000000 000000
 2     SW1:    CLP points to: 0
 3     Last Updated CCW: 000000 000000    WC=0.  ADR=0
```

COMPAR

Compare data buffer. The data compare is done based on the word count set up earlier by a SET LEN command. Normally a data compare could be done after running some microcode which did a data transfer. No data comparisons are done unless the diagnostic is specifically requested to do so. Example:

```
DEBUG> COMPAR

Data Compare Error:
Word     --Correct Data--       ---Actual Data--
    2       525252 525252          400000 000001
    4       525252 525252          400000 000001
    6       525252 525252          400000 000001
Words in error = 64.
```

10.5.3.14 Switches

SWITCH

Enter switch information. Switches can be entered as in the following example. Usually, though, it is easier just to type the switch name to turn on or off a switch.

```
DEBUG> SWITCH

Print the selectable pgm switches? (Y or N) - Y

----Left Side Switches--------   ----Right Side Switches--------

SW   KL10     11    Function      SW   Value  Function
--   ----    ----   --------      --   -----  --------
 0  400000  100000  Abort         18  400000  Program trace
 ...

Switches = 020010 004000

20010  LH Switches <# or ?> - ?

Type  Y,N,^,CR,^Z

N ABORT -
N RSTART -
...

4000  RH Switches <# or ?> - ?

Type  Y,N,^,CR,^Z

N TRACE -
N INHFLT -
...
```

SWPRIN

Print current switch settings. Example:

```
DEBUG> SWPRIN

Switches:  020010 002000   PNTLPT OPRSEL UDEBUG
```

```
NOPNT    RELIAB   TRACE    DSPEAR   UDEBUG
PNTLPT   TXTINH   INHFLT   NIPORT   MDEBUG
DING     OPRSEL   INHMSG   CIPORT   SDEBUG
LOOPER            LOOPGM   MPROC    LDEBUG
ERSTOP            LOOPTS   MCBUS
PALERS            RUNALL
```

The name of each of these switches can be typed. The result is to complement the switch setting. If left hand switches are typed in exec mode an error message is printed.

11.0 PROGRAM RUN TIME

The run time varies according to the sequence of tests chosen, the test options selected, and the program switch options selected.

In default mode the run time is approximately 3 minutes per port per pass, excluding any time taken to print error messages. The diagnostic runs slightly faster in exec mode. If the "OPRSEL" switch is not used, each port will run 5 passes, bringing the total run time to around 28 minutes in user mode (24 minutes in exec mode).

Running all of the tests with "RELIAB" switch set extends the time by a factor of 4, to around 2 hours.

```
---------------------------------------------------------------------------
VERSION:        MAINDEC-10-DFPTA-A      Version 0.1

AUTHOR:         Richard Stockdale

DATE:           December 31, 1983

REASON:         Original release.
---------------------------------------------------------------------------
VERSION:        MAINDEC-10-DFPTA-B      Version 0.2

AUTHOR:         Richard Stockdale

DATE:           December 31, 1984

REASON:         Added support for TOPS10 usermode operation.
```

VERSION: MAINDEC-10-DFPTA-C Version 0.3

UPDATE AUTHOR: Gregory A. Scott

DATE: May, 1985

REASONS:

Added title statements to each module for DDT typeout of module that this symbol is defined in.

Fix TOPS-10 usermode operation: Don't check TOPS-20 privs on the -10. Use proper DIAG. UUO function. Trust setting of MONTYP rather than try and figure it out again. Print "Exec Only Test" rather than "Ommitted". Add proper TOPS-10 error codes to T10TAB. Use monitor uptime instead of time of day for timing operations.

Change strange output of help messages (using several ASCII psuedo-ops padded by spaces to word boundries followed by an ASCIZ, depends on literal pool not being rearranged by MACRO) to big ASCIZ string, add .DIRECTIVE FLBLST to clean listing.

Fix error message if port not available to indicate to operator what command is needed to free the port for diagnostic use.

If DFPTA was running the one port on one job, and another DFPTA wanted to run the other port, the second copy of DFPTA would "hang" trying to determine what ports existed. This was because the monitor makes a second diagnostic wait for the port rather than returning an error like "another job owns the port". Now if the switches CIPORT and NIPORT are on, we onlu look for that type of port, enabling paralell operation of both CI and NI.

VERSION: MAINDEC-10-DFPTA-D Version 0.4

UPDATE AUTHOR: Gregory A. Scott

DATE: July 1, 1985

REASONS:

If run without OPRSEL on the diagnostic would exit before releasing the port to the operating system. If any diagnostic is run later which tried to use the port it would "hang" trying to get the port again.

Output of "End pass" messages was incorrect. Always print them rather than trying to print them based on various conditions.

Output "?" in front of error message for batch runs.

DOCUMENT
****
DFPTA LST
****

COPYRIGHT 1985
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASS. 01754

TABLE OF CONTENTS

```
1
982     *PARAM* CONSOLE DATA SWITCH ASSIGNMENTS, SEPT 18,1979
1017    *PARAM* PROGRAM/SUBROUTINE PARAMETERS, SEPT 18,1979
1223    ERROR HANDLING UUO DEFINITIONS, SEPT 18,1979
1261    *FIXED* FIXED CONTROL AND DISPATCH STORAGE, SEPT 18,1979
1589    IPA REGISTER STORAGE
3       Externs / Entrys
126     Initialization
269     Test/Command Selection
1595    Test Dispatching
2632    Port Debugger
6804    Program Utility Routines
7635    SPEAR Reporting
7832    Test Execute Routines
10066   Miscellaneous Test Variables
2       EBUS Module Tests
7511    EBUS Module Error Printing Routines
2       MP Control Module:  2910 Basic Instruction Tests
2       EBUS/MPROC 2901 Tests (Part 1)
2       EBUS/MPROC 2901 Tests (Part 2)
31      Register Interference Tests
2       EBUS/MPROC 2901 Tests (Part 3)
```

TABLE OF CONTENTS

```
27      Register Interference Tests
2       EBUS/MPROC 2901 Tests (Part 4)
2       MPROC Module Tests
7116    Miscellaneous Routines
2       CBUS MODULE TESTS
2       CBUS MODULE TESTS
2       Port Controller Utility Package
60      Port Register Read/Write Routines
383     Port Specific Utility Routines
776     PI System Routines
1179    Page Fail Handling Routines
1265    Channel Handling Routines
1970    User Mode Support Routines
2455    Miscellaneous Routines
2       UTILITY Routines Module
66      Printout/Typin Routines
311     Clock Handling Routines
591     TOPS10/20 Related Routines
667     File Handling Routines
850     Data Pattern Generator/Compare Routines
1029    Data Patterns
1301    Switch Handling Routines
```

TABLE OF CONTENTS

1821 Miscellaneous Routines
2514 Scope Looping And Error Reporting Routine

```
        *****************************************************************
   1    DFPTA TITLE FILE
        *****************************************************************

        27              Copyright (C) 1985
                        DIGITAL EQUIPMENT CORPORATION, MAYNARD MASS.

                        This software is furnished under a license  for  use  only
                        on a single computer system and  may  be  copied only with
                        the inclusion of the of the above copyright  notice.  This
                        software, or any other copies thereof, may not be provided
                        or otherwise made available to any other person except for
                        use  on such system and to one who agrees to these license
                        terms. Title to and ownership of the software shall at all
                        times remain in DEC.

                        The information in this  document  is  subject  to  change
                        without notice and should not be construed as a commitment
                        by Digital Equipment Corporation.

                        DEC assumes  no  responsibility for the use or reliability
                        of its software on equipment which is not supplied by DEC.

                        Author:         Dick Stockdale
                        Maintainer:     36-Bit Diagnostic Engineering

        51          Program Parameters

        67          Assembly Parameters

        98          Channel Logout Status Bits

        114         Right Hand Switches

        135         Macro Definitions

        139         Text macros

        166         ASCII text macro

        172         RPUT/RGET - Multiple put/get

        204         Test descriptor entry generator macros

        217         Network Definitions

        293         SSTABLE - Single step table entry generator macro

        316         ATABLE - Single step table entry generator macro (with EBUF checking)
```

```
             333     TTABLE - Test execute table entry generator macro
             351     CTABLE - Data test execute table entry generator macro
             368     ITABLE - Interrupt test execute table entry generator macro
             386     Test Dispatch Table Definitions
             409     Port CONI/CONO CSR Bit Assignments
             442     OPDEF Definitions
             462     Bit Definitions For "SCOPE" UUO Handler - Error Message Printer
             475     Bit Definitions For Microcode executer routine (MEXEC)
             487     Bit Definitions For Test executer routine (TEXEC)
             504     Bit Definitions For Data Test executer routine (CEXEC)
             521     Bit Definitions For Interrupt Test executer routine (IEXEC)
             538     Microword Field Definitions
             938     Microword Macros (Note: If ADDR not specified, 'ADDR'+1 is used).
         ****************************************************************************
     982 *PARAM* CONSOLE DATA SWITCH ASSIGNMENTS, SEPT 18,1979
         ****************************************************************************

             987    DATA SWITCHES (READ FROM CONSOLE IN EXEC MODE OR TYPED IN IN USER MODE)
                    LEFT HALF SWITCHES ARE PRE-ASSIGNED FOR SUBROUTINE PACKAGE USE
                    AND CONTROL LOOPING, PRINTING (TTY OR OTHER DEVICE) AND MISC. FUNCTIONS

         ****************************************************************************
    1017 *PARAM* PROGRAM/SUBROUTINE PARAMETERS, SEPT 18,1979
         ****************************************************************************

            1020    SPECIAL SUBPROGRAM LINKAGES

            1031    SPECIAL SUBROUTINE FATAL HALTS
                    USED TO REPORT ERRORS THAT CAUSE THE SUBROUTINES TO BE UNUSABLE
```

```
      ******************************************************************
1223  ERROR HANDLING UUO DEFINITIONS, SEPT 18,1979
      ******************************************************************

      ******************************************************************
1261  *FIXED* FIXED CONTROL AND DISPATCH STORAGE, SEPT 18,1979
      ******************************************************************

      ******************************************************************
1589  IPA REGISTER STORAGE
      ******************************************************************

      ******************************************************************
3     Externs / Entrys
      ******************************************************************

      ******************************************************************
126   Initialization
      ******************************************************************

      ******************************************************************
269   Test/Command Selection
      ******************************************************************

      272     DISPAT - Routine to dispatch to proper test as typed by the operator
                       in response to the 'WHAT TEST' prompt.

      508     INDLIS - List of classes that may be specified with individual test numbers

      520     DISLIS -- List of all valid test names that can be typed

      569     DISGO - Test dispatch addresses

      663     MBUS - Execute MBUS related tests

      706     ALL     Execute all tests
              EBUS    Execute EBus module tests
              SEQ     Execute SEQ related tests
              ALU     Execute 2901 related tests
              MPROC   Execute MPROC module tests
              MBUS    Execute MBUS related tests
              CBUS    Execute CBus module tests

      739     ISOPNT - Print fault isolation data

      833     Network Data Areas

      921     TSTLIS - List test description for all tests.
```

```
          968     TSTLST - List test description for either single test or class of tests.

          1044    TSTDIA - Return to DIAMON

          1055    TSTDDT - Enter DDT

          1070    TSTHLT - Program halts - CONTINUE will start it back up.

          1084    CONFIG - Determine/print device configuration

          1163    CONPNT - Print configuration

          1194    TSTSEL - Select device(s) to test

          1353    TSTHLP - Help for 'WHAT TEST' question.

          1437    TSTDSA - Disable a test.
                  TSTENA - Enable a test.

          1484    TSTDSL - List disabled tests

          1514    TSTSWI/TSTSWP - Input switches / Print switch selections

          1531    TSTSSW - Switch complement commands

          1543    TSTTAK - Take commands from a file
                  TSTTAN - Take commands from a file without echoing results

          *****************************************************************
  1595    Test Dispatching
          *****************************************************************

          1598    TSTSPC - Run only one test

          1625    RANGEN - Random number generator

          1680    TSTDIS - Dispatch to the tests

          1927    Test Dispatch List - in sequential order

          2187    Test Dispatch List - in order for fault isolation

          2488    FLTHAN - Fault handler

          2523    FLTCHK - See if this test should fail based on possible modules list

          2559    TSTDEC - Decode test specified as XXnnn where XX is the test class
                           and nnn is the test number.
```

```
        *******************************************************************
 2632   Port Debugger
        *******************************************************************

        2635     TSTDEB - Console command utility for the port.
        2938     .DBCMD - List of acceptable commands
        3093     .DBDIS - List of dispatch addresses
        3250     DHELP - Print DEBUG> help message
        3419     DSET - Set/clear/print options
        3466     DSETAD - Set start address for START/SSTEP/CONT commands
        3492     DSETCS - Set start CSR data for START/CONT commands
        3511     DSETPR - Set/clear automatic parity generation for load cram command
        3535     DSETEB - Set/clear preserve EBUF for single stepping
        3558     DSETLN - Set length of data transfer
        3583     DSETDA - Set data pattern to insert in buffer
        3626     DSETFN - Set default microcode file name
        3656     DSETWH - Print all selected options
        3674     DSETFA - Set failing network
        3718     DSETPL - List data patterns
        3763     DSETHE - Set help message
        3792     Program Flags
        3807     DECSR - Examine CSR register
        3821     DDCSR - Deposit CSR register
        3836     DZCSR - Zero CSR register
        3848     DCONI - Loop on CONI's
        3861     DCONO - Loop on CONO's
        3878     DDATI - Loop on DATAI's
```

3891 DDATO - Loop on DATAO's
3907 DROUTN - Routine to do special diagnostic loop
3936 DRESET - Issue an EBUS Reset
3947 DCLEAR - Issue a 'Port Clear'
3957 DSINIT - Set up start addr/data/history prior to single step command
3973 DSTART - Start the port
4018 DSTOP - Stop the port
4048 DSSTEP - Single step the port
DTRACE - Single step the port and trace progress
4148 DSSPNT - Print single step history
4172 DSSCLR - Clear single step history data
4193 PNTHST - Print a single step history entry
4216 SAVHST - Save single step history data
4239 DCONT - Continue the port
4267 DEEBUF - Examine EBUF register
4289 DDEBUF - Deposit EBUF register
DZEBUF - Zero EBUF register
4316 DELAR - Examine LAR register
4337 DDRAR - Deposit RAR register
4355 DZRAR - Zero RAR register
4368 DECRAM - Examine CRAM locations specified by 'PARG1' and 'PARG2'.
4376 DDCRAM - Change CRAM locations.
4384 DBCRAM - Change CRAM locations by field
4392 DACRAM - Alter CRAM locations.
4400 DZCRAM -- Zero CRAM addresses specified.
4408 DLCRAM -- List CRAM addresses specified.

4995 DEALU - Examine 2901 registers
5097 DDALU - Deposit register
5174 DZALU - Zero 2901 registers
5285 DELOCS - Examine local storage
5388 DDLOCS - Deposit local storage
5460 DZLOCS - Zero local storage
5569 DMARK - Set mark bit in CRAM locations
5612 DRMARK - Remove mark bit in CRAM locations
5655 DCMARK - Clear all mark bits
5728 DLMARK - List locations with mark bit set
5814 DENEXT - Examine next
5879 DDNEXT - Deposit next
5930 DESTAK - examine 2910 stack
5983 DBREAK - Set breakpoint
6034 DRBRK - Remove breakpoint
6073 DCBRK - Clear all breakpoints
6141 DLBRK - List breakpoints
6230 DLOAD - Load microcode into CRAM from a test.
6265 DLIST - List microcode from test
6294 DFLOAD - Load microcode from file into CRAM
6330 DFVERF - Verify microcode from file versus CRAM
6392 DFLIST - List microcode in file
6421 DFEXAM - Examine microcode in file
6450 DTRANS - Translate CSR bits to English
6465 DCCODE - Print condition codes

```
        6558    DFILLN - Fill CRAM with special data (J=PC+1,OPCode)
        6585    DFILLP - Fill CRAM with special data (J=PC,OPCode)
        6609    DRDINI - Initialize everything for a read
        6636    DWRINI - Initialize everything for a write
        6646    DBPNT - Print buffer contents
        6679    DCWPNT - Print CCW list
        6690    DLGPNT - Print logout data
        6704    DCOMPA - Compare data buffer
        6724    DSWIT - Switch complement commands
        6738    IPRINT - Print interrupt history
        6794    IINIT - Initialize interrupt system

        *****************************************************************
6804    Program Utility Routines
        *****************************************************************

        6807    TRACE - Program Trace Routine
        6876    TSTGET - Get test related data based on test dispatch address
        6915    SELTST - Select a test to run.
        6947    CHKTST - Check if this is a valid test to execute and set up test type.
        6992    TSTPNT - Print test name/number
        7014    ODELAY - Delay a specified amount of time (in milliseconds in AC0)
        7050    DDELAY - Delay a specified amount of time (in milliseconds in AC0)
        7077    MLOADN - Load microcode and verify it (do not print errors)
                MLOADY - Load microcode and verify it (print errors)
        7208    MVPNT - Print verify error data
        7257    MLIST - List microcode of a test
        7293    PNTCRM - Print CRAM word
```

```
          7376     PNTWD - Print a CRAM word (xxxx xxxx xxxx xxxx xxxx)

          7400     CALPAR - Calculate parity for a CRAM location

          7449     TLOAD - Load microcode/verify it for a test

          7497     CSRPNT - Print CSR register as 'CSR: xxxxxx,,xxxxxx' followed by
                            English translation.

          7515     CSRENG - Print CSR data in English

          7547     CSRENB - CSR bits

          7594     SAVCRM - Save specified CRAM locations in a buffer

          7612     RESCRM - Restore CRAM locations

          ***************************************************************
7635      SPEAR Reporting
          ***************************************************************

          7638     SPREP1 - Print initial SPEAR report 'diagnostic started'

          7691     SPREP2 - Print error or successful completion SPEAR report

          7820     GETPPN - Find out PPN number

          ***************************************************************
7832      Test Execute Routines
          ***************************************************************

          7835     SEXEC - Execute entries in a 'Single Step Table'

          8048     SSPNT - Print out single step data that resulted in an error

          8095     AEXEC - ALU type test execute routine

          8226     AAPNT - Print results message

          8282     BEXEC - Execute entries in a 'Single Step Table', verifying EBUF data

          8516     BBPNT - Print out single step data that resulted in an error

          8591     TEXEC - Test execute routine

          8851     TTPNT - Print out error data

          8911     CEXEC - Data test execute routine
```

```
                9282    CCPNT - Print out error data
                9373    IEXEC - Interrupt test execute routine
                9891    IIPNT - Print out error data

                ******************************************************************
         10066  Miscellaneous Test Variables
                ******************************************************************

                10118   Data Buffer Areas For Reading And Writing
                10127   Data transfer paramters
                10147   Special code to test SPEAR entries
                10187   Special code to produce an EXE version

                ******************************************************************
         2      EBUS Module Tests
                ******************************************************************

                48      TEST 1 - Is there an port out there?
                97      Test 2 - RH20/Port Interaction
                201     Test 3 - RH20/Port Interaction
                302     Test 4 - RH20/Port Interaction
                406     Test 5 - RH20/Port Interaction
                506     Test 6 - CSR Cleared After Reset
                568     Test 7 - Read/Write CSR Bits
                733     TEST 10 - CSR Bits Interaction
                833     TEST 11 - EBUF Access Test
                887     TEST 12 - EBUF/CSR Conflict Test
                948     TEST 13 - KMUX Parity Generators
                1092    TEST 14 - EBUF/CSR Conflict Test
                1176    TEST 15 - EBUF Reset Test
                1241    TEST 16 - EBUF Data Path Test
```

1400 TEST 17 - CSR Bit 18 Self Clearing Test
1450 TEST 20 - CSR Bit 18 Reset Test
1501 TEST 21 - CSR Bit 18 Reset Test
1559 TEST 22 - Read/Write EBUF When Ucode Running
1654 TEST 23 - Par Gen - Not Forcing an Error
1839 TEST 24 - Par Gen - Forcing an Error
1929 TEST 25 - Read only CSR Bits
2031 TEST 26 - CSR 01 'DIAG RQST CSR'
2117 TEST 27 - CSR 01 'DIAG RQST CSR'
2207 TEST 30 - CSR 01 'DIAG RQST CSR'
2291 TEST 31 - CSR 01 'DIAG RQST CSR'
2378 TEST 32 - CSR 01 'DIAG RQST CSR'
2457 TEST 33 - CSR 01 'DIAG RQST CSR'
2541 TEST 34 - CSR 02 'DIAG CSR CHNG'
2643 Test 35 - Constant MUX
2877 Test 36 - Constant MUX Interference
2965 TEST 37 - CSR Bit 24 - EBUS PE Occurs at Wrong Time
3022 TEST 40 - EBUS Transfer via DATAO's
3145 TEST 41 - EBUS Transfer via DATAI's
3301 TEST 42 - CSR 05 'Rqst Interrupt'
3379 TEST 43 - CSR 05 'Rqst Interrupt'
3455 TEST 44 - CSR 05 'Rqst Interrupt'
3531 TEST 45 - CSR 06 'CRAM Par Err'
3605 TEST 46 - CSR 07 'MBUS Err'
3685 TEST 47 - CSR 11 'Idle' Bit

3907 TEST 50 - CSR 12 'Disable Complete' Bit
4128 TEST 51 - CSR 13 'Enable Complete' Bit
4347 TEST 52 - x
4359 TEST 53 - MPROC Run Bit
4436 TEST 54 - Basic KMUX Parity Test
4528 TEST 55 - Clearing MPROC Run Bit
4611 TEST 56 - Clearing MPROC Run Bit
4689 TEST 57 - Verify 'Diag Single Cyc'
4765 TEST 60 - PI Level n/0 Interrupts
4949 TEST 61 - Check PIA Level 0
5046 TEST 62 - Check Interrupt on Level 1-7
5173 TEST 63 - CSR 05 'Rqst Interrupt' Clearing
5293 TEST 64 - IOP Function 0 - 40+2N Interrupt
5565 TEST 65 - IOP Function 1 - 40+2N Interrupt
5836 TEST 66 - x
5849 TEST 67 - x
5861 TEST 70 - IOP Function 4 - Examine
6120 TEST 71 - IOP Function 5 - Deposit
6385 TEST 72 - IOP Function 7 - Examine/Increment
6642 TEST 73 - CSR 25-26 & 28 KL-side
6845 TEST 74 - CSR 25-26 & 28 Port-side
7067 TEST 75 - CSR 27 & 30-31 KL-side
7233 TEST 76 - CSR 27 & 30-31 Port-side

```
        ***************************************************************
7511    EBUS Module Error Printing Routines
        ***************************************************************

        7607     PFPNT - Print page fail data

        7625     INTPNT - Print interrupt activity

        ***************************************************************
2       MP Control Module:  2910 Basic Instruction Tests
        ***************************************************************

        35       Test 1 - JMAP Instruction

        205      Test 2 - JZ Instruction

        320      Test 3 - JZ Instruction - Effect on Reg/Ctr

        443      Test 4 - JZ Instruction - Effect on Reg/Ctr

        566      Test 5 - CONT Instruction

        716      Test 6 - CJP Instruction

        808      Test 7 - CJV Instruction

        899      Test 10 - CJS Instruction

        990      Test 11 - CJS Instruction

        1076     Test 12 - CJS Instruction

        1169     Test 13 - CRTN Instruction

        1251     Test 14 - CRTN Instruction

        1368     Test 15 - Stack location #1

        1533     Test 16 - Stack location #2

        1698     Test 17 - Stack location #3

        1865     Test 20 - Stack location #4

        2028     Test 21 - Stack location #5

        2195     Test 22 - Increment/Decrement Stack Pointer

        2286     Test 23 - Increment/Decrement Stack Pointer
```

```
2380    Test 24 - Increment/Decrement Stack Pointer
2475    Test 25 - Increment/Decrement Stack Pointer
2572    Test 26 - Increment/Decrement Stack Pointer
2671    Test 27 - x
2681    Test 30 - CJPP Instruction
2780    Test 31 - CJPP Instruction
2904    Test 32 - CJPP Instruction
2994    Test 33 - LDCT Instruction
3071    Test 34 - LDCT Instruction
3175    Test 35 - Register/Counter
3405    Test 36 - JRP Instruction
3510    Test 37 - JSRP Instruction
3626    Test 40 - JSRP Instruction
3743    Test 41 - JSRP Instruction
3833    Test 42 - PUSH Instruction
3953    Test 43 - PUSH Instruction
4060    Test 44 - PUSH Instruction
4156    Test 45 - LOOP Instruction
4267    Test 46 - LOOP Instruction
4375    Test 47 - LOOP Instruction
4478    Test 50 - RFCT Instruction
4780    Test 51 - RFCT Instruction
4880    Test 52 - RFCT Instruction
5068    Test 53 - RFCT Instruction
5441    Test 54 - RPCT Instruction
```

```
                5559    Test 55 - RPCT Instruction
                5808    Test 56 - TWB Instruction
                5908    Test 57 - TWB Instruction
                5993    Test 60 - TWB Instruction
                6084    Test 61 - TWB Instruction
                6193    Test 62 - TWB Instruction
                6300    Test 63 - TWB Instruction
                6384    Test 64 - Stack Interference Tests
                6532    Test 65 - Stack Interference Tests
                6677    Test 66 - Reg/Ctr Interference Tests
                6853    Test 67 - Reg/Ctr Interference Tests
                7007    TEST 70 - Full Speed Sequencer Test

                ************************************************************
        2       EBUS/MPROC 2901 Tests (Part 1)
                ************************************************************

                48      Test 1 - 2901 0's Test
                292     Test 2 - 2901 1's Test
                650     Test 3 - 2901 OR Test - D OR 0
                750     Test 4 - 2901 AND Test - D AND 0
                840     Test 5 - 2901 NOT_R_AND_S Test - D NOT_RS 0
                930     Test 6 - 2901 XOR Test - D XOR 0
                1029    Test 7 - 2901 XNOR Test - D XNOR 0
                1128    Test 10 - 2901 R+S Test - D + 0
                1227    Test 11 - 2901 R-S Test - D - 0
                1326    Test 12 - 2901 S-R Test - 0 - D
                1429    Test 13 - 2901 Writing/Reading Registers
                1833    Test 14 - 2901 Q-Register Test
                1930    Test 15 - 2901 Register Interference Test
                2135    Test 16 - 2901 OR Test
                2386    Test 17 - 2901 AND Test
                2620    Test 20 - 2901 NOT_R_AND_S Test
                2866    Test 21 - 2901 XOR Test
                3154    Test 22 - 2901 XNOR Test
                3521    Test 23 - 2901 R + S Test, No Carry
                3848    Test 24 - 2901 R + S Test, With Carry
                4179    Test 25 - 2901 R - S Test - Without Carry
                4497    Test 26 - 2901 R - S Test - With Carry
                4805    Test 27 - 2901 S - R Test - Without Carry
                5121    Test 30 - 2901 S - R Test - With Carry
                5441    ADDLOG - Routine to logically add 2 numbers

                *****************************************************************
        2       EBUS/MPROC 2901 Tests (Part 2)
                *****************************************************************

                *****************************************************************
        31      Register Interference Tests
                *****************************************************************

                34      Test 31 - 2901 Q-Register Interference Test

                *****************************************************************
        2       EBUS/MPROC 2901 Tests (Part 3)
                *****************************************************************
```

```
          *****************************************************************
   27     Register Interference Tests
          *****************************************************************

          30        TEST 32 - 2901 RAM Register Interference Test

          *****************************************************************
   2      EBUS/MPROC 2901 Tests (Part 4)
          *****************************************************************

          32        TEST 33 - Q-Register Shift
          200       TEST 34 - Q-Register Right/Left Shift
          358       TEST 35 - RAM Register Shift
          1025      TEST 36 - RAM Data Movement Test
          1384      TEST 37 - RAM/Q-Register Data Movement Test
          1583      TEST 40 - RAM Data Movement/Shifting Test
          1782      TEST 41 - RAM/Q-Reg Data Movement/Shifting Test
          1978      TEST 42 - 2901 Register / Constant Mux Test

          *****************************************************************
   2      MPROC Module Tests
          *****************************************************************

          46        TEST 1 - CRAM Data Test (RAR12=0)
          208       TEST 2 - CRAM Test (RAR12=1)
          303       TEST 3 - CRAM Addressing Test
          391       TEST 4 - RAR/LAR Data Path Test - LDCRAM
          506       TEST 5 - RAR/LAR Data Path Test - RDCRAM
          626       TEST 6 - Cram Ctrl Register Test
          875       TEST 7 - Cram Ctrl Register Test
          1124      TEST 10 - MBUS Error Test (MPROC module only)
          1243      TEST 11 - MBUS Error Test (CBUS module only)
          1337      TEST 12 - MBUS Error Test (MPROC module only)
```

```
            1482      TEST 13 - MBUS Error Test (CBUS module only)
            1587      TEST 14 - CCCbusAvail Test
            1880      TEST 15 - CCGrntCSR Test
            2138      TEST 16 - CCFEQ0 Test
            2722      TEST 17 - CCCSRChng Test
            2946      TEST 20 - CCEbParErr Test
            3124      TEST 21 - CCRcvrBufAFul Test
            3316      TEST 22 - CCRcvrBufBFul Test
            3508      TEST 23 - CCXmtrAttn Test
            3700      TEST 24 - CCEbusRqst Test
            3912      TEST 25 - CCIntrActive Test
            4116      TEST 26 - CCMBSign Test
            4313      TEST 27 - CCMVParChk Test
            4611      TEST 30 - CCCbusParErr Test
            4852      TEST 31 - CCPliParErr Test
            5179      TEST 32 - CCChanErr Test
            5462      TEST 33 - CCCbLstWd Test
            5695      TEST 34 - Basic Local Store Test
            5827      TEST 35 - Local Store Data Test
            6556      TEST 36 - Local Store Address Test
            6760      TEST 37 - Local Store RAM Mode Test
            6872      Test 40 - Jump Mux Test
            6974      TEST 41 - CRAM Control Register Parity Generators Test
```

```
                 *****************************************************************
        7116     Miscellaneous Routines
                 *****************************************************************

                 *****************************************************************
        2        CBUS MODULE TESTS
                 *****************************************************************

                 42        Test 1 - Fmtr Data Loopback
                 622       Test 2 - PLI Buffer Data Loopback
                 837       Test 3 - CBUS to EBUF Data Transfer
                 978       Test 4 - 2901 to CBUS Data Transfer
                 1134      Test 5 - CBUS to 2901 Multiple word Transfer
                 1463      Test 6 - 2901 to CBUS Multiple word Transfer
                 1830      Test 7 - CBUS to EBUS Data Transfer
                 2114      Test 10 - EBUS to CBUS Data Transfer
                 2414      Test 11 - Fmtr Cleared by 'Port Clear'
                 2492      Test 12 - Fmtr Basic Up Shifting Test
                 2607      Test 13 - Fmtr Up Shift 4 Test - BUS Ctl=Cbus
                 2918      Test 14 - Fmtr Up Shift 4 Test - BUS Ctl=Fmtr
                 3228      Test 15 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=0
                 3553      Test 16 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=0
                 3876      Test 17 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=1
                 4199      Test 20 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=1
                 4523      Test 21 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=0
                 4811      Test 22 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=0
                 5098      Test 23 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=1
                 5384      Test 24 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=1
                 5673      Test 25 - Fmtr Basic Down Shifting Test
```

```
            5787     Test 26 - Fmtr Down Shift 4 Test
            6106     Test 27 - Fmtr Down Shift 4 (from PLIN) Test - SELRHTNIB=0
            6434     Test 30 - Fmtr Down Shift 4 (from PLIN) Test - SELRHTNIB=0
            6760     Test 31 - Fmtr Down Shift 4 (from PLIN) Test - SELRHTNIB=1
            7088     Test 32 - Fmtr Down Shift 4 (from PLIN) Test - SELRHTNIB=1
            7416     Test 33 - Fmtr Down Shift 8 (from PLIN) Test - SELRHTNIB=0
            7708     Test 34 - Fmtr Down Shift 8 (from PLIN) Test - SELRHTNIB=0
            7999     Test 35 - Fmtr Down Shift 8 (from PLIN) Test - SELRHTNIB=1
            8289     Test 36 - Fmtr Down Shift 8 (from PLIN) Test - SELRHTNIB=1
            8578     Test 37 - Fmtr Ring Buffer Test
            ****************************************************************
      2     CBUS MODULE TESTS
            ****************************************************************
            34       Test 40 - CMUX Selected by PLINTOCMUX
            226      Test 41 - CBUF Loaded Correctly
            466      Test 42 - PMUX Selected by MPCBUFTOPLOUT
            577      Test 43 - PMUX Selected by MPCBUFTOPLOUT/MPZEROLFTNIB
            688      Test 44 - DMUX Selected Correctly
            916      Test 45 - CBUS Parity Checking - CBUS to CBUS Module
            1121     Test 46 - CBUS Parity Checking - CBUS Module to CBUS
            1287     Test 47 - CBUS Parity Checking - CBUS to CBUS Module
            1533     Test 50 - T Field Timing Test
            1674     Test 51 - CBUS Store
            1903     Test 52 - CBUS Input Buffer
            1989     Test 53 - Parity Predictor - CBIN to Fmtr/MPINDSTCOMP=0
            2332     Test 54 - Parity Predictor - CBIN to Fmtr/MPINDSTCOMP=1
```

```
               2661    Test 55 - Parity Predictor - Fmtr to PLOUT
               2957    Test 56 - Parity Predictor - PLIN to Fmtr/MPCOREDUMP=0
               3257    Test 57 - Parity Predictor - PLIN to Fmtr/MPCOREDUMP=1
               3569    Test 60 - Parity Predictor - Fmtr to CBOUT
               ***************************************************************
       2       Port Controller Utility Package
               ***************************************************************

               ***************************************************************
       60      Port Register Read/Write Routines
               ***************************************************************
               63      Notes On I/O Routine Design
               113     Port Register Load Routines
               125     LDEBUF - Load EBUF
               142     LDCSR - Load CSR (This is simply a CONO).
               156     LDRAR - Load RAR
               180     LDCRAM - Load CRAM
               199     Port Register Read Routines
               216     RDEBUF - Read EBUF
               233     RDCSR - Read CSR
               251     RDLAR - Read LAR
               275     RDCRAM - Read CRAM
               293     .CONI - Do a simple CONI (no error checking)
               313     .CONO - Do a simple CONO
               333     .DATAI - Do a simple DATAI (no error checking)
               353     .DATAO - Do a simple DATAO
               372     ERESET - Issue an EBUS reset
```

```
        *****************************************************************
383     Port Specific Utility Routines
        *****************************************************************

        386     ISTOP - Stop IPA either by a port clear or by clearing the CSR register

        431     IPACLR - Issue a port clear

        442     SETEBU - Check that 'Diag Test EBUF' bit is set in CSR and set if not.

        462     SETLAR - Check that 'Diag Sel LAR' bit is set in CSR and set if not.

        481     CLREBU - Check that 'Diag Test EBUF' is not set in CSR and clear if is.

        500     CHKCSR - Check if error bits are set in the CSR register

        529     IPASRT - Start the port at the address given in 'SADDR' and with
                         initial parameters given in 'SDATA'.

        609     IPASTP - Stop the port

        673     IPASST - Single step the port at the address given in 'SNEXT' and
                         with initial parameters given in 'SDATA'.

        *****************************************************************
776     PI System Routines
        *****************************************************************

        779     INITPI - Initialize and Turn On PI System

        820     .PIOFF - Detach port from PI system

        836     .PION - Attach port to PI system (Channel 4)

        854     PISYOF - Turn PI system off

        863     PISYON - Turn PI system on

        871     SETVEC - Set up interrupt vectors for an port under test

        895     SET2N - Set up 40 + 2N locations with interrupt dispatch instruction

        917     VIINT - Vectored interrupt (Port)

        982     NVIINx - Non-Vectored interrupt (Port)

        1132    .INPNT - Print special interrupt data
```

```
         *****************************************************************
  1179   Page Fail Handling Routines
         *****************************************************************

         1182     PFSET - Set up EPT locations 500-502 to handle page fails

         1215     PFCLR - Restore EPT locations 500-502 to original values

         1229     PFCHK - Check reason for page fail and dispatch appropriately

         *****************************************************************
  1265   Channel Handling Routines
         *****************************************************************

         1268     Routines To Interface With A Channel

         1304     CHINIT - Channel Initialization Routine

         1360     GENCCW - Channel Command Word Generator

         1508     GRINDF - Subroutine to build a forward transfer CCW list.

         1534    ******** Next 2 instructions are ommitted for DFPTA tests. ****************
                 ******** This causes every CCW to start at same memory location. **********

         1559     CCWST - RH20 CCW Store Routine

         1585     GETLOG - Snapshot The Channel Logout Area

         1621     LOGPNT - Channel Logout Printer

         1693     CCWPNT - Channel Command List Printer

         1781     CHP20 - Builds a TOPS20 User Mode Channel Program

         1844     CHP10 - Builds a TOPS10 Channel Command List

         1896     DIAGER - Report Monitor Call Error In User Mode

         1918     Some Channel Routine Variables

         1949     CHDATA - Obtain channel logout data and inspect results
         ******************************************************************
1970     User Mode Support Routines
         ******************************************************************

         1973      DSETUP - Setup Routine For Diagnostic Function

         2037      CPUTST - Determine Which Of 2 CPU's To Run On

         2078      DEVREL - User Mode Release Routine

         2206      DEVREQ - User Mode Request Routine

         2359      DGEPNT - Decode and print DIAG error

         2408      USRION/USRIOF - Turn User I/O Privileges On/Off

         2437      ENABLE - Enable Capabilities of This Process

         ******************************************************************
2455     Miscellaneous Routines
         ******************************************************************

         2458      FFF - Routine To Handle Fatals

         ******************************************************************
2        UTILITY Routines Module
         ******************************************************************

         ******************************************************************
66       Printout/Typin Routines
         ******************************************************************

         69        CONVSX - Convert sixbit to octal

         109       ALTCHH - Handle altmode intercept

         134       CLRBUF - Clear input buffer

         149       TTYYES - Answer yes or no question

         177       PSDN - Prints signed decimal numbers

         202       .PNTOC - Print n digit octal number with leading zeros suppressed

         240       PPDEC - Print decimal number in a field of size x.

         276       PPDECF - Force print decimal number in a field of size x.
```

```
              *****************************************************************
      311     Clock Handling Routines
              *****************************************************************

              314      STCLOK - Initialize program runtime clock

              354      MONTIM - Read program runtime from monitor

              393      PTIME - Print time of day

              411      RUNTME - Calculate and print program runtime

              517      PSTAMP - Check for an 'S' typed.

              559      LSTAMP - Check for an 'S' typed.

              *****************************************************************
      591     TOPS10/20 Related Routines
              *****************************************************************

              594      .RESET - Subroutine to issue a RESET JSYS/UUO

              610      .CLOSE - Subroutine to close opened files

              638      .EXIT - Subroutine to exit job in user mode

              *****************************************************************
      667     File Handling Routines
              *****************************************************************

              670      FINIT - Initialize reading of microcode file

              696      CHRIN - Routine to read an ASCII character from the selected file.

              706      CHRINS - Routine to read an ASCII character from the selected file.

              733      FFAIL - Routine to report '"DIAMON"' open failure on a selected file.

              751      FGETW - Get microword entry from file

              797      FGETA - Obtain microcode load address 'ADDR:' from file

              824      FGETD - Obtain microcode data entry from file (4 digits).
```

```
        ******************************************************************
 850    Data Pattern Generator/Compare Routines
        ******************************************************************

        853      BUFGEN - Generate the specified data pattern in the data buffer.
                 BUFCOM - Compare the data buffer with expected data

        989      DATMES - A data comparison error has been found.  Report the error,
                 printing a maximum of three lines of errors.

        ******************************************************************
 1029   Data Patterns
        ******************************************************************

        1032     Data Patterns

        1209     PATPNT - Translate pattern number to description

        ******************************************************************
 1301   Switch Handling Routines
        ******************************************************************

        1304     .SWCHP - Print switch settings in English

        1370     SWITT - Get value of switches

        1383     .ISWT - Complete switch handling

        1476     .SLEFT - Input left hand switches

        1539     .SRIGT - Enter right hand switches

        1708     SWCHPT - Prints current state of switches

        1720     SWCOM - Switch complement routine (AC1 has switch command number 0-n)

        ******************************************************************
 1821   Miscellaneous Routines
        ******************************************************************

        1824     PARSER - Additional CPU parity error service

        1871     .COMM - Match a command entered against a command table.

        1919     .COMM2 - Compare command typed to table to see if valid

        1951     .SARG - Obtain SIXBIT argument

        2037     .OARG - Obtain octal argument
```

```
          2126    DECYN - Decode an argument which should be NO or YES
          2152    .DARG - Obtain octal/decimal argument
          2244    CHKARG - Check for argument when there shouldn't be any
          2289    LASARG - Check for last argument when there shouldn't be any more
          2330    FIOFF - Routine to turn off the 'file input' switch
          2348    FINCMD - Input a sixbit command from a file
          2395    FSARG - Input a sixbit argument from a file
          2442    FOARG - Input a octal argument from a file
          2503    FEOL - Skip characters until end of line is reached

          ******************************************************************
     2514 Scope Looping And Error Reporting Routine
          ******************************************************************

          2517    SCOPE1 - Error reporter and scope loop controller
```

```
                                        SUBTTL  DFPTA TITLE FILE

                                        SALL

                                        UNIVERSAL DFPTA
                                        SEARCH  MONSYM,MACSYM

                                        EXTERN  START,STARTA,PGMNAM,SCOPE1
                                        EXTERN  PPDEC,PPDECF,ALTCHH,.PNTOC

                                        .REQUIRE DFPTA1
                                        .REQUIRE DFPTA2
                                        .REQUIRE DFPTA3
                                        .REQUIRE DFPTA4
                                        .REQUIRE DFPTA5
                                        .REQUIRE DFPTA6
                                        .REQUIRE DFPTA7
                                        .REQUIRE DFPTA8
                                        .REQUIRE DFPTA9
                                        .REQUIRE DFPTAA
                                        .REQUIRE DFPTAI
                                        .REQUIRE DFPTAU

                000000                  MCNVER==0
                000004                  DECVER==4

                                ;*      Copyright (C) 1985
                                ;*      DIGITAL EQUIPMENT CORPORATION, MAYNARD MASS.
                                ;*
                                ;*      This software is furnished under a license  for  use  only
                                ;*      on a single computer system and  may  be  copied only with
                                ;*      the inclusion of the of the above copyright  notice.  This
                                ;*      software, or any other copies thereof, may not be provided
                                ;*      or otherwise made available to any other person except for
                                ;*      use  on such system and to one who agrees to these license
                                ;*      terms. Title to and ownership of the software shall at all
                                ;*      times remain in DEC.
                                ;*
                                ;*      The information in this  document  is  subject  to  change
                                ;*      without notice and should not be construed as a commitment
                                ;*      by Digital Equipment Corporation.
                                ;*
                                ;*      DEC assumes  no  responsibility for the use or reliability
                                ;*      of its software on equipment which is not supplied by DEC.
                                ;*
                                ;*      Author:         Dick Stockdale
                                ;*      Maintainer:     36-Bit Diagnostic Engineering

```

```
                                        ;#*******************************************************************
                                        ;* Program Parameters
                                        ;#*******************************************************************

                                        ; AC Assignments.

                        000014                  PAT=14                          ; data pattern in use
                        000015                  ERFLG=15                        ; error flag (gets cleared by SCOPER)
                        000016                  MBCN=16                         ; channel 54000,,0 to 57400,,0
                        000017                  P=17                            ; stack pointer

                                        ; Size Of Channel Control Word Buffer

                        000315                  CBUFSZ==^D205                   ; 200 words (enough for 102400 words)


                                        ;#*******************************************************************
                                        ;* Assembly Parameters
                                        ;#*******************************************************************

                        000005                  ITERAT==5                       ; default to 5 passes
                        000001                  KL10==1                         ; assemble for KL10
                        000001                  EXCASB==1                       ; assemble for exec mode
                        000001                  USRASB==1                       ; assemble for user mode
                        000001                  MEMMAP==1                       ; allow memory to be mapped

                        000001                  $LPAPER==1                      ; don't list literals

                        030000                  SADR1=BEGIN
                        030000                  SADR2=BEGIN
                        030000                  SADR3=BEGIN
                        030007                  SADR4=SRTDDT
                        000000*                 SADR5=START                     ; ^C restart
                        030007                  SADR6=SRTDDT
                254200  000000*                 SADR7=HALT START
                254200  000000*                 SADR8=HALT START
                254200  000000*                 SADR9=HALT START
                254200  000000*                 SADR10=HALT START
                254200  000000*                 SADR11=HALT START

                711523  655207                  PAREA1==711523,,655207          ; random number base
                        000000                  PAREA2==0
                444660  644100                  PAREA3==SIXBIT/DFPTA/
                605664  000000                  PAREA4==SIXBIT/PNT/
                        000000                  PAREA5==0
                        000000                  PAREA6==0
```

```

                                        ;#*************************************************************
                                        ;* Channel Logout Status Bits
                                        ;#*************************************************************

        400000  000000                          LOGO==1B0                       ; channel log out bit
        200000  000000                          MPER==1B1                       ; memory parity error
        100000  000000                          ADRPE==1B2                      ; set when no address parity error
        040000  000000                          NOTWC0==1B3                     ; not word count zero
        020000  000000                          NEXM==1B4                       ; non-existant memory
        000400  000000                          LAXER==1B9                      ; last xfer error
        000200  000000                          IPAER==1B10                     ; IPA20 error
        000100  000000                          LGWC==1B11                      ; long word count
        000040  000000                          SHWC==1B12                      ; short word count
        000020  000000                          OVN==1B13                       ; overrrun


                                        ;#*********************************************************************
                                        ;* Right Hand Switches
                                        ;#*********************************************************************

                400000                          TRACET==1B18                    ; program test # trace feature
                200000                          INHFLT==1B19                    ; inhibit fault isolation
                100000                          INHMSG==1B20                    ; inhibit error message printout
                040000                          LOOPGM==1B21                    ; loop on program
                020000                          LOOPTS==1B22                    ; loop on test
                010000                          RUNALL==1B23                    ; run all segments of a test
                004000                          DSPEAR==1B24                    ; disable SPEAR switch
                002000                          DEBUG1==1B25                    ; debug mode (debug without a port)
                001000                          DEBUG2==1B26                    ; debug mode (print error messages)
                000400                          DEBUG3==1B27                    ; debug mode (print single step data)
                000200                          DEBUG4==1B28                    ; debug mode (don't load test ucode)
                000100                          DEBUG5==1B29                    ; debug mode (for fault isolation)
                000040                          NIPORT==1B30                    ; NI port
                000020                          CIPORT==1B31                    ; CI port
                000002                          MMPROC==1B34                    ; missing Mproc module
                000001                          MCBUS==1B35                     ; missing Cbus module
```

```

;#***************************************************************************
;* Macro Definitions
;#***************************************************************************

;#**************************************************************************
;* Text macros
;#**************************************************************************

        DEFINE  TMSG(TEXT),<PNTMSG      [ASCIZ ^TEXT^]>

        DEFINE  TMSGC(TEXT),<PNTMSG     [ASCIZ ^
TEXT^]>

        DEFINE  TMSGD(TEXT),<PNTMSG     [ASCIZ ^TEXT
^]>

        DEFINE  TMSGCD(TEXT),<PNTMSG    [ASCIZ ^
TEXT
^]>
        DEFINE  FMSG(TEXT),<PNTMSF      [ASCIZ ^TEXT^]>

        DEFINE  FMSGC(TEXT),<PNTMSF     [ASCIZ ^
TEXT^]>

        DEFINE  FMSGD(TEXT),<PNTMSF     [ASCIZ ^TEXT
^]>

        DEFINE  FMSGCD(TEXT),<PNTMSF    [ASCIZ ^
TEXT
^]>

;#**************************************************************************
;* ASCII text macro
;#**************************************************************************

        DEFINE  TASCIZ (TEXT)<[ASCIZ /TEXT/]>
```

```
;*********************************************************************
;* RPUT/RGET - Multiple put/get
;*********************************************************************

        DEFINE  RPUT (R0,R1<-1>,R2<-1>,R3<-1>,R4<-1>,R5<-1>,R6<-1>,R7<-1>,R8<-1>,R9<-1>)<
        PUT     R0
        XLIST
        IFGE R1,<PUT    R1>
        IFGE R2,<PUT    R2>
        IFGE R3,<PUT    R3>
        IFGE R4,<PUT    R4>
        IFGE R5,<PUT    R5>
        IFGE R6,<PUT    R6>
        IFGE R7,<PUT    R7>
        IFGE R8,<PUT    R8>
        IFGE R9,<PUT    R9>
        LIST>

        DEFINE  RGET (R0,R1<-1>,R2<-1>,R3<-1>,R4<-1>,R5<-1>,R6<-1>,R7<-1>,R8<-1>,R9<-1>)<
        GET     R0
        XLIST
        IFGE R1,<GET    R1>
        IFGE R2,<GET    R2>
        IFGE R3,<GET    R3>
        IFGE R4,<GET    R4>
        IFGE R5,<GET    R5>
        IFGE R6,<GET    R6>
        IFGE R7,<GET    R7>
        IFGE R8,<GET    R8>
        IFGE R9,<GET    R9>
        LIST>
```

```

                                ;#*********************************************************************
                                ;* Test descriptor entry generator macros
                                ;#*********************************************************************

             000003                     TENLEN==3                       ; length of test descriptor entry

                                        DEFINE  TEST    (TADD,TMASK,TMCODE,TDESC)<
                                        XLIST
                                        TMASK!TADD
                                        TMCODE,,[ASCIZ ^TDESC^]
                                        LIST>


                                ;#*********************************************************************
                                ;* Network Definitions
                                ;#*********************************************************************

             000001                     E1==1
             000002                     E2==2
             000003                     E3==3
             000004                     E4==4
             000005                     E5==5
             000006                     E6==6
             000007                     E7==7
             000010                     E8==^D8
             000011                     E9==^D9
             000012                     E10==^D10
             000013                     E11==^D11
             000014                     E12==^D12
             000015                     E13==^D13
             000016                     E14==^D14
             000017                     E15==^D15
             000020                     E16==^D16
             000021                     E17==^D17
             000022                     E18==^D18
             000023                     E19==^D19
             000024                     E20==^D20
             000025                     E21==^D21
             000026                     E22==^D22
             000027                     E23==^D23
             000030                     E24==^D24
```

```

                000031           M1==^D25
                000032           M2==^D26
                000033           M3==^D27
                000034           M4==^D28
                000035           M5==^D29
                000036           M6==^D30
                000037           M7==^D31
                000040           M8==^D32
                000041           M9==^D33
                000042           M10==^D34
                000043           M11==^D35
                000044           M12==^D36
                000045           M13==^D37
                000046           M14==^D38
                000047           M15==^D39
                000050           M16==^D40
                000051           M17==^D41
                000052           M18==^D42
                000053           M19==^D43

                000054           C1==^D44
                000055           C2==^D45
                000056           C3==^D46
                000057           C4==^D47
                000060           C5==^D48
                000061           C6==^D49
                000062           C7==^D50
                000063           C8==^D51
                000064           C9==^D52
                000065           C10==^D53
                000066           C11==^D54
                000067           C12==^D55
                000070           C13==^D56
                000071           C14==^D57
                000072           C15==^D58
                000073           C16==^D59
                000074           C17==^D60
                000075           C18==^D61
                000076           C19==^D62
                000077           C20==^D63
                000100           C21==^D64
                000101           C22==^D65
                000102           C23==^D66
                000103           C24==^D67

        400000  000000           MLAST==1B0
```

```
;#*********************************************************************
;* SSTABLE - Single step table entry generator macro
;
; SSLAST - No arguments used
; SSSTRT - Arguments:   ARG1 - # ssteps
;                       ARG2 - Start address
;                       ARG3 - End address
; SSCONT - Arguments:   ARG1 - # ssteps
;                       ARG2 - End address
; SSCALL - Argument:  ARG1 - Routine address
; SSCHK -- Argument:  ARG1 - Routine address
; SSJRST - Argument:  ARG1 - Table address
;#*********************************************************************

        DEFINE  STABLE  (CMD,ARG1,ARG2,ARG3)<
        IFE CMD,<0>                             ; SSLAST
        IFE CMD-1,<1B2!ARG1_^D24!ARG2_^D12!ARG3>; SSTRT
        IFE CMD-2,<2B2!ARG1_^D24!ARG2>          ; SSCONT
        IFE CMD-3,<3B2!ARG1>                    ; SSCALL
        IFE CMD-4,<4B2!ARG1>                    ; SSCHK
        IFE CMD-5,<5B2!ARG1>>                   ; SSJRST


;#*********************************************************************
;* ATABLE - Single step table entry generator macro (with EBUF checking)
;
; This is same as STABLE except that an additional word is used to
; save the correct EBUF contents.
;#*********************************************************************

        DEFINE  ATABLE  (CMD,ARG1,ARG2,ARG3,ARG4<0>)<
        IFE CMD,<0>                             ; SSLAST
        IFE CMD-1,<1B2!ARG1_^D24!ARG2_^D12!ARG3>; SSTRT
        IFE CMD-2,<2B2!ARG1_^D24!ARG2>          ; SSCONT
        IFE CMD-3,<3B2!ARG1>                    ; SSCALL
        IFE CMD-4,<4B2!ARG1>                    ; SSCHK
        IFE CMD-5,<5B2!ARG1>                    ; SSJRST
        IFE CMD-1,<ARG4>
        IFE CMD-2,<ARG4>>
```

```

;#**********************************************************************
;* TTABLE - Test execute table entry generator macro
;
; This is same as STABLE except for different command definitions.
;#**********************************************************************

        DEFINE  TTABLE  (CMD,ARG1,ARG2<0>)<
        IFE CMD,<0>                      ; TLAST
        IFE CMD-1,<1B3!ARG1>             ; TSTART
        IFE CMD-2,<2B3!ARG1>             ; TCONT
        IFE CMD-3,<3B3!ARG1>             ; TCALL
        IFE CMD-4,<4B3!ARG1>             ; TCALLC
        IFE CMD-5,<5B3!ARG1_^D18!ARG2>   ; TCHECK
        IFE CMD-6,<6B3!ARG1>             ; TJRST
        IFE CMD-7,<7B3!ARG1>             ; TEXIT
        IFE CMD-10,<10B3!ARG1>>          ; TSSTAR


;#**********************************************************************
;* CTABLE - Data test execute table entry generator macro
;#**********************************************************************

        DEFINE  CTABLE  (CMD,ARG1,ARG2)<
        IFE CMD,<0>                      ; CLAST
        IFE CMD-1,<1B3!ARG1_^D18!ARG2>   ; CSETRD
        IFE CMD-2,<2B3!ARG1_^D18!ARG2>   ; CSETWR
        IFE CMD-3,<3B3!ARG1_^D18!ARG2>   ; CSETRW
        IFE CMD-4,<4B3!ARG1>             ; CSTART
        IFE CMD-5,<5B3!ARG1>             ; CCALL
        IFE CMD-6,<6B3!ARG1>             ; CCALLC
        IFE CMD-7,<7B3!ARG1_^D18!ARG2>   ; CWAIT
        IFE CMD-10,<10B3>                ; CCOMP
        IFE CMD-11,<11B3!ARG1>           ; CJRST
        IFE CMD-12,<12B3>>               ; CEXIT

;#**********************************************************************
;* ITABLE - Interrupt test execute table entry generator macro
;#**********************************************************************

        DEFINE  ITABLE  (CMD,ARG1,ARG2)<
        IFE CMD,<0>                      ; ILAST
        IFE CMD-1,<1B3!ARG1_^D18!ARG2>   ; ISETEX
        IFE CMD-2,<2B3!ARG1_^D18!ARG2>   ; ISETDE
        IFE CMD-3,<3B3!ARG1_^D18!ARG2>   ; ISETID
        IFE CMD-4,<4B3!ARG1_^D18!ARG2>   ; ISETIN
        IFE CMD-5,<5B3!ARG1_^D18!ARG2>   ; ISETEI
        IFE CMD-6,<6B3!ARG1>             ; ICALL
        IFE CMD-7,<7B3!ARG1>             ; ICALLC
        IFE CMD-10,<10B3!ARG1>           ; IWAIT
        IFE CMD-11,<11B3!ARG1>           ; IJRST
        IFE CMD-12,<12B3!ARG1>           ; ISTART
        IFE CMD-13,<13B3>>               ; IEXIT
```

```

                                 ;#*********************************************************************
                                 ;* Test Dispatch Table Definitions
                                 ;#*********************************************************************

     400000  000000                      EBUS==1B0                     ; EBus tests
     200000  000000                      MPROC==1B1                    ; Mproc tests
     100000  000000                      CBUS==1B2                     ; CBus tests
     040000  000000                      SEQ==1B3                      ; 2910 tests
     020000  000000                      ALU==1B4                      ; 2901 tests
     010000  000000                      MBUS==1B5                     ; MBUS tests
     770000  000000                      ALL==77B5                     ; All tests

             000000                      ZEBUS==0B17                   ; EBUS class test
     000001  000000                      ZSEQ==1B17                    ; SEQ class test
     000002  000000                      ZALU==2B17                    ; ALU class test
     000003  000000                      ZMPROC==3B17                  ; MPROC class test
     000004  000000                      ZCBUS==4B17                   ; CBUS class test

     001000  000000                      TUSER==1B8                    ; cannot run test in user mode
     000400  000000                      NDMP==1B9                     ; need Mproc module to run
     000200  000000                      NDCB==1B10                    ; need Mproc & CBus module to run
     000100  000000                      TDENA==1B11                   ; test dispatch enable (0-yes,1-no)
```

```

                                  ;#*********************************************************************
                                  ;* Port CONI/CONO CSR Bit Assignments
                                  ;#*********************************************************************

        400000  000000                  PORTP==1B0                       ; port present
        200000  000000                  CSRRQS==1B1                      ; diag request CSR
        100000  000000                  CSRCHN==1B2                      ; diag CSR changed
        040000  000000                  DINIT==1B3                       ; diag initialized
        010000  000000                  RQINT==1B5                       ; request interrupt
        004000  000000                  CRAMPE==1B6                      ; CRAM parity error
        002000  000000                  MBERR==1B7                       ; MBUS error
        000100  000000                  IDLE==1B11                       ; IDLE loop
        000040  000000                  DCOMP==1B12                      ; disable complete
        000020  000000                  ECOMP==1B13                      ; enable complete
                400000                  PCLEAR==1B18                     ; port clear
                200000                  TSTEBF==1B19                     ; diag test EBUF
                100000                  GENEPE==1B20                     ; diag generate EBUS parity error
                040000                  SELLAR==1B21                     ; diag select LAR/SQR
                020000                  SINCYC==1B22                     ; diag single cycle
                010000                  SPARE1==1B23                     ; spare bit
                004000                  EBUSPE==1B24                     ; EBUS parity error
                002000                  FQUERR==1B25                     ; free queue error
                001000                  MVRERR==1B26                     ; data mover error
                000400                  CMDQAV==1B27                     ; command queue available
                000200                  RESQAV==1B28                     ; response queue available
                000100                  SPARE2==1B29                     ; spare bit
                000040                  PDSABL==1B30                     ; disable
                000020                  PENABL==1B31                     ; enable
                000010                  MPRUN==1B32                      ; mproc run
                000004                  PIA00==1B33                      ; PIA field bit 0
                000002                  PIA01==1B34                      ; PIA field bit 1
                000001                  PIA02==1B35                      ; PIA field bit 2
```

```

                                   ;#********************************************************************
                                   ;* OPDEF Definitions
                                   ;#********************************************************************

 104000 000000                              OPDEF   JSYS    [104000,,0]      ; TOPS-20 command
 104000 000100                              OPDEF   CFIBF   [JSYS 100]       ; clear TOPS-20 input buffer

 001000 000000                              OPDEF   PNTOCC  [1B8]            ; print n digits leading 0's suppressed
 002000 000000                              OPDEF   PDECF   [2B8]            ; force print decimal number
 006000 000000                              OPDEF   PDEC    [6B8]            ; print decimal number
 007000 000000                              OPDEF   ALTCHK  [7B8]            ; check for altmode character typed
 027000 000000                              OPDEF   SCOPER  [27B8]           ; error handler UUO

        000000*                             LUUO1==.PNTOC                    ; service routine for n digit octal
        000000*                             LUUO2==PPDECF                    ; force print decimal number
        000000*                             LUUO6==PPDEC                     ; print decimal number
        000000*                             LUUO7==ALTCHH                    ; check if an altmode was typed
        000000*                             LUUO27=SCOPE1                    ; scope loop/error print routine



                                   ;#********************************************************************
                                   ;* Bit Definitions For "SCOPE" UUO Handler - Error Message Printer
                                   ;#********************************************************************

 100000 000000                              MSG==1B2                         ; ASCIZ message
 200000 000000                              CALL==2B2                        ; call special print routine
 300000 000000                              GOTO==3B2                        ; goto (new address of error table)
 040000 000000                              TXNOT==1B3                       ; print/call entry if TXTINH not set
 020000 000000                              TXYES==1B4                       ; print/call entry if TXTINH set
 060000 000000                              TXALL==3B4                       ; print/call entry always
 010000 000000                              LAST==1B5                        ; last table entry


                                   ;#********************************************************************
                                   ;* Bit Definitions For Microcode executer routine (MEXEC)
                                   ;#********************************************************************

        000000                              SSLAST==0                        ; last entry
        000001                              SSSTRT==1                        ; single step with start addr given
        000002                              SSCONT==2                        ; single step without start addr given
        000003                              SSCALL==3                        ; call a routine
        000004                              SSCHK== 4                        ; call a routine and exit + 1 if test
                                                                             ;   is complete, + 2 if not.
        000005                              SSJRST==5                        ; go to new table location
```

```

                                  ;#*********************************************************************
                                  ;* Bit Definitions For Test executer routine (TEXEC)
                                  ;#*********************************************************************

             000000                     TLAST==0                ; last entry
             000001                     TSTART==1               ; start port at a particular address
             000002                     TCONT==2                ; continue from where stopped
             000003                     TCALL==3                ; call routine
             000004                     TCALLC==4               ; call routine and conditionally
                                                                ;   exit test
             000005                     TCHECK==5               ; check CSR for completion code
             000006                     TJRST==6                ; go to new table location
             000007                     TEXIT==7                ; exit test (if any error yet)
             000010                     TSSTAR==10              ; start port at a particular address
                                                                ;   without doing a port clear first


                                  ;#*********************************************************************
                                  ;* Bit Definitions For Data Test executer routine (CEXEC)
                                  ;#*********************************************************************

             000000                     CLAST==0                ; last entry
             000001                     CSETRD==1               ; set up read data transfer
             000002                     CSETWR==2               ; set up write data transfer
             000003                     CSETRW==3               ; set up read/write data transfer
             000004                     CSTART==4               ; start port at particular address
             000005                     CCALL==5                ; call routine
             000006                     CCALLC==6               ; call routine/conditionally exit test
             000007                     CWAIT==7                ; wait for DT completion
             000010                     CCOMP==10               ; compare data buffer
             000011                     CJRST==11               ; go to new table location
             000012                     CEXIT==12               ; exit test (if any error yet)


                                  ;#*********************************************************************
                                  ;* Bit Definitions For Interrupt Test executer routine (IEXEC)
                                  ;#*********************************************************************

             000000                     ILAST==0                ; last entry
             000001                     ISETEX==1               ; set up examine
             000002                     ISETDE==2               ; set up deposit
             000003                     ISETID==3               ; set up increment/decrement
             000004                     ISETIN==4               ; set up interrupt
             000005                     ISETEI==5               ; set up examine/increment
             000006                     ICALL==6                ; call routine
             000007                     ICALLC==7               ; call routine/conditionally exit test
             000010                     IWAIT==10               ; wait for interrupt completion
             000011                     IJRST==11               ; go to new table location
             000012                     ISTART==12              ; start port at particular address
             000013                     IEXIT==13               ; exit test (if any error yet)
```

```
;#**********************************************************************
;* Microword Field Definitions
;
;  These definitions are arranged for the macro which generates pairs
;  of microwords as (addr & parity flags & left 20 bits,right 36 bits).
;  These are of the form:
;
;  Word 1:  Bits 00-11 - Microword load address (0000-7777)
;                12-35 - Bits 0-23 of microword to load
;  Word 2:  Bits 00-34 - Bits 24-57,59 of microword to load
;                35 ---- Force Bad Parity flag
;
;  In detail:
;
;  Word 1:  Bits 00-11 - Microword load address
;                12-23 - MWJMPFLD
;                24    - MWPAR
;                25    - MWOUTPUTENA
;                26-35 - MWMGCFLD
;
;  Word 2:  Bits 00-02 - MWSORCEFLD
;                03-05 - MWFUNCTFLD
;                06-08 - MWDESTFLD
;                09    - MWCCENA
;                10    - MWRAMODE
;                11-14 - MWPORTAFLD
;                15-18 - MWPORTBFLD
;                19-23 - MWSKIPFLD
;                24-26 - MWBUSCTLFLD
;                27    - MWCARRY
;                28-31 - MWCTRLFLD
;                32    - MWTIMEFLD
;                33    - MWSPARE00
;                34    - MWMARKBIT
;
;                35    - Force bad parity flag
;
;  The microwords are generated using a set of field  definitions
;  describing how the two words are to be constructed. Each field
;  is defined as a bit position in the first word, a bit position
;  in the right word, and a magnitude describing the size  of the
;  field.  These are defined as follows:
;
;      .Lxxxx - Specifies the rightmost bit position of the field
;               in the left word. If the field does not appear in
;               left word, this will be zero.
;
;      .Rxxxx - Specifies the rightmost bit position of the field
;               in the right word.  If the field  does not appear
;               in right word, this will be zero.
;
;      .Mxxxx - Specifies the maximum magnitude of the  field  in
;               the word.
```

```
                                        ;  The microwords are generated as follows:
                                        ;
                                        ;  Left word = ((value AND .Mxxxx) times .Lxxxx) ! next field ...
                                        ;  Right word = ((value AND .Mxxxx) times .Rxxxx) ! next field ...
                                        ;
                                        ;  Parity is calculated prior to loading into the CRAM.  The parity
                                        ;  bit  may be specified explicitly by specifying EVEN or ODD.  The
                                        ;  microcode  loader  will  ensure  that the word has the specified
                                        ;  parity.  If not specified, the correct parity will be calculated.
                                        ;
                                        ;  Sample microword definitions:
                                        ;
                                        ;  (a) Specify:  J 1000, OUTENA, SORCE fld = 2, DEST fld = 3
                                        ;                load address = 120
                                        ;
                                        ;      Type:     MWORD  <ADDR=120,J=1000,OENA,SORC=2,DEST=3>
                                        ;#*****************************************************************

        000100  000000                        .LADDR==1B11                ; MW load address
                000000                        .RADDR==0
                007777                        .MADDR==7777

                010000                        .LJ==1B23                   ; MW jump field         (00:11)
                000000                        .RJ==0
                007777                        .MJ==7777

                004000                        .LPAR==1B24                 ; MW parity             (12:12)
                000000                        .RPAR==0
                000001                        .MPAR==1

                002000                        .LOENA==1B25                ; MW output enable      (13:13)
                000000                        .ROENA==0
                000001                        .MOENA==1

                000001                        .LMGC==1B35                 ; MW magic number field (14:23)
                000000                        .RMGC==0
                001777                        .MMGC==1777

                000000                        .LSORC==0                   ; MW source field       (24:26)
        100000  000000                        .RSORC==1B2
                000007                        .MSORC==7

                000000                                .LSAQ==0            ; Source = AQ
                000000                                .RSAQ==0B2
                000007                                .MSAQ==7

                000000                                .LSAB==0            ; Source = AB
        100000  000000                                .RSAB==1B2
                000007                                .MSAB==7

```

```
   641                 000000                            .LSOQ==0               ; Source = OQ
   642         200000  000000                            .RSOQ==2B2
   643                 000007                            .MSOQ==7
   644
   645                 000000                            .LSOB==0               ; Source = OB
   646         300000  000000                            .RSOB==3B2
   647                 000007                            .MSOB==7
   648
   649                 000000                            .LSOA==0               ; Source = OA
   650         400000  000000                            .RSOA==4B2
   651                 000007                            .MSOA==7
   652
   653                 000000                            .LSDA==0               ; Source = DA
   654         500000  000000                            .RSDA==5B2
   655                 000007                            .MSDA==7
   656
   657                 000000                            .LSDQ==0               ; Source = DQ
   658         600000  000000                            .RSDQ==6B2
   659                 000007                            .MSDQ==7
   660
   661                 000000                            .LSDO==0               ; Source = DO
   662         700000  000000                            .RSDO==7B2
   663                 000007                            .MSDO==7
   664
   665                 000000                    .LFUNC==0                      ; MW function field     (27:29)
   666         010000  000000                    .RFUNC==1B5
   667                 000007                    .MFUNC==7
   668
   669                 000000                            .LPLUS==0              ; R PLUS S function
   670                 000000                            .RPLUS==0B5
   671                 000007                            .MPLUS==7
   672
   673                 000000                            .LSMIN==0              ; R MINUS S function
   674         010000  000000                            .RSMIN==1B5
   675                 000007                            .MSMIN==7
   676
   677                 000000                            .LRMIN==0              ; R MINUS S function
   678         020000  000000                            .RRMIN==2B5
   679                 000007                            .MRMIN==7
   680
   681                 000000                            .LOR==0                ; R OR S function
   682         030000  000000                            .ROR==3B5
   683                 000007                            .MOR==7
   684
   685                 000000                            .LAND==0               ; R AND S function
   686         040000  000000                            .RAND==4B5
   687                 000007                            .MAND==7
   688
```

```
                000000                              .LNAND==0                  ; NOT R AND S function
        050000  000000                              .RNAND==5B5
                000007                              .MNAND==7

                000000                              .LXOR==0                   ; R XOR S function
        060000  000000                              .RXOR==6B5
                000007                              .MXOR==7

                000000                              .LXNOR==0                  ; R XNOR S function
        070000  000000                              .RXNOR==7B5
                000007                              .MXNOR==7

                000000                      .LD==0                             ; MW destination field  (30:32)
        001000  000000                      .RD==1B8
                000007                      .MD==7

                000000                      .LCENA==0                          ; MW CC enable          (33:33)
        000400  000000                      .RCENA==1B9
                000001                      .MCENA==1

                000000                              .LDISA==0                  ; MW CC disable
                000000                              .RDISA==0B9
                000001                              .MDISA==1

                000000                              .LENA==0                   ; MW CC enable
        000400  000000                              .RENA==1B9
                000001                              .MENA==1

                000000                      .LRAM==0                           ; MW RAM mode           (34:34)
        000200  000000                      .RRAM==1B10
                000001                      .MRAM==1

                000000                      .LA==0                             ; MW Port A field       (35:38)
        000010  000000                      .RA==1B14
                000017                      .MA==17

                000000                      .LB==0                             ; MW Port B field       (39:42)
                400000                      .RB==1B18
                000017                      .MB==17

                000000                      .LSK==0                            ; MW skip field         (43:47)
                010000                      .RSK==1B23
                000037                      .MSK==37

                000000                              .LCCON==0                  ; Select CC 'on' (CCCSRCHNG)
                030000                              .RCCON==03B23
                000037                              .MCCON==37

```

```
   737                  000000                  .LCCOF==0               ; Select CC 'off' (CCGRNTCSR)
   738                  010000                  .RCCOF==01B23
   739                  000037                  .MCCOF==37
   740
   741                  000000                  .LCCCA==0               ; Select CCCBUSAVAIL
   742                  000000                  .RCCCA==0B23
   743                  000037                  .MCCCA==37
   744
   745                  000000                  .LCCGC==0               ; Select CCGRNTCSR
   746                  010000                  .RCCGC==01B23
   747                  000037                  .MCCGC==37
   748
   749                  000000                  .LCCFZ==0               ; Select CCFEQ0
   750                  020000                  .RCCFZ==02B23
   751                  000037                  .MCCFZ==37
   752
   753                  000000                  .LCCCC==0               ; Select CCCSRCHNG
   754                  030000                  .RCCCC==03B23
   755                  000037                  .MCCCC==37
   756
   757                  000000                  .LCCEB==0               ; Select CCEBPARERR
   758                  040000                  .RCCEB==4B23
   759                  000037                  .MCCEB==37
   760
   761                  000000                  .LCCAF==0               ; Select CCRCVRBUFAFUL
   762                  050000                  .RCCAF==5B23
   763                  000037                  .MCCAF==37
   764
   765                  000000                  .LCCBF==0               ; Select CCRCVRBUFBFUL
   766                  060000                  .RCCBF==6B23
   767                  000037                  .MCCBF==37
   768
   769                  000000                  .LCCXA==0               ; Select CCXMTRATTN
   770                  070000                  .RCCXA==7B23
   771                  000037                  .MCCXA==37
   772
   773                  000000                  .LCCER==0               ; Select CCEBUSRQST
   774                  100000                  .RCCER==10B23
   775                  000037                  .MCCER==37
   776
   777                  000000                  .LCCIA==0               ; Select CCINTRACTIVE
   778                  110000                  .RCCIA==11B23
   779                  000037                  .MCCIA==37
   780
   781                  000000                  .LCCMB==0               ; Select CCMBSIGN
   782                  120000                  .RCCMB==12B23
   783                  000037                  .MCCMB==37
   784
```

```
    785              000000                          .LCCMP==0          ; Select CCMVRPARCHK
    786              130000                          .RCCMP==13B23
    787              000037                          .MCCMP==37
    788
    789              000000                          .LCCCP==0          ; Select CCCBUSPARERR
    790              140000                          .RCCCP==14B23
    791              000037                          .MCCCP==37
    792
    793              000000                          .LCCPP==0          ; Select CCPLIPARERR
    794              150000                          .RCCPP==15B23
    795              000037                          .MCCPP==37
    796
    797              000000                          .LCCCE==0          ; Select CCCHANERR
    798              160000                          .RCCCE==16B23
    799              000037                          .MCCCE==37
    800
    801              000000                          .LCCLW==0          ; Select CCCBLSTWD
    802              170000                          .RCCLW==17B23
    803              000037                          .MCCLW==37
    804
    805              000000                          .LLSAD==0          ; Load SAD Reg
    806              200000                          .RLSAD==20B23
    807              000037                          .MLSAD==37
    808
    809              000000                          .LSKMB==0          ; Select MBUS field
    810              210000                          .RSKMB==21B23
    811              000037                          .MSKMB==37
    812
    813              000000                          .LLDLM==0          ; Load Local Mem
    814              230000                          .RLDLM==23B23
    815              000037                          .MLDLM==37
    816
    817              000000                          .LRDLM==0          ; Read Local Mem
    818              220000                          .RRDLM==22B23
    819              000037                          .MRDLM==37
    820
    821              000000                          .LSKCN==0          ; Select constant field
    822              240000                          .RSKCN==24B23
    823              000037                          .MSKCN==37
    824
    825              000000                 .LBUS==0                    ; MW bus control field  (48:50)
    826              001000                 .RBUS==1B26
    827              000007                 .MBUS==7
    828
    829              000000                          .LSELP==0          ; Select PLI field
    830              001000                          .RSELP==1B26
    831              000007                          .MSELP==7
    832
    833              000000                          .LSELM==0          ; Select Mbus field
    834              002000                          .RSELM==2B26
    835              000007                          .MSELM==7
    836
```

```
        000000                   .LSELF==0               ; Select Fmtr field
        003000                   .RSELF==3B26
        000007                   .MSELF==7

        000000                   .LSELC==0               ; Select Cbus field
        004000                   .RSELC==4B26
        000007                   .MSELC==7

        000000                   .LSELE==0               ; Select Ebus field
        005000                   .RSELE==5B26
        000007                   .MSELE==7

        000000           .LCRY==0                        ; MW carry              (51:51)
        000400           .RCRY==1B27
        000001           .MCRY==1

        000000           .LOP==0                         ; MW control field      (52:55)
        000020           .ROP==1B31
        000017           .MOP==17

        000000                   .LJZ==0                 ; Jump zero instruction
        000000                   .RJZ==00B31
        000017                   .MJZ==17

        000000                   .LCJS==0                ; Conditional JSB pipeline instruction
        000020                   .RCJS==01B31
        000017                   .MCJS==17

        000000                   .LJMAP==0               ; Jump MAP instruction
        000040                   .RJMAP==02B31
        000017                   .MJMAP==17

        000000                   .LCJP==0                ; Conditional jump pipeline instruction
        000060                   .RCJP==03B31
        000017                   .MCJP==17

        000000                   .LPUSH==0               ; Push/cond load counter instruction
        000100                   .RPUSH==04B31
        000017                   .MPUSH==17

        000000                   .LJSRP==0               ; Cond JSB R/Pipeline instruction
        000120                   .RJSRP==05B31
        000017                   .MJSRP==17

        000000                   .LCJV==0                ; Conditional jump vector instruction
        000140                   .RCJV==06B31
        000017                   .MCJV==17

```

```
   885            000000                  .LJRP==0            ; Cond jump R/Pipeline instruction
   886            000160                  .RJRP==07B31
   887            000017                  .MJRP==17
   888
   889            000000                  .LRFCT==0           ; Repeat loop, counter#0 instruction
   890            000200                  .RRFCT==10B31
   891            000017                  .MRFCT==17
   892
   893            000000                  .LRPCT==0           ; Repeat pipeline, counter#0 instr
   894            000220                  .RRPCT==11B31
   895            000017                  .MRPCT==17
   896
   897            000000                  .LCRTN==0           ; Conditional RTN instruction
   898            000240                  .RCRTN==12B31
   899            000017                  .MCRTN==17
   900
   901            000000                  .LCJPP==0           ; Cond jump pipeline and pop instr
   902            000260                  .RCJPP==13B31
   903            000017                  .MCJPP==17
   904
   905            000000                  .LLDCT==0           ; Load counter and continue instruction
   906            000300                  .RLDCT==14B31
   907            000017                  .MLDCT==17
   908
   909            000000                  .LLOOP==0           ; Test end loop instruction
   910            000320                  .RLOOP==15B31
   911            000017                  .MLOOP==17
   912
   913            000000                  .LCONT==0           ; Continue instruction
   914            000340                  .RCONT==16B31
   915            000017                  .MCONT==17
   916
   917            000000                  .LTWB==0            ; Three way branch instruction
   918            000360                  .RTWB==17B31
   919            000017                  .MTWB==17
   920
   921            000000              .LTIM==0                ; time field
   922            000010              .RTIM==1B32
   923            000001              .MTIM==1
   924
   925            000000              .LSPAR==0               ; 1st spare bit
   926            000004              .RSPAR==1B33
   927            000001              .MSPAR==1
   928
   929            000000              .LMARK==0               ; mark bit
   930            000002              .RMARK==1B34
   931            000001              .MMARK==1
   932
   933            000000              .LBAD==0                ; force bad parity
   934            000001              .RBAD==1B35
   935            000001              .MBAD==1
```

```
;#*********************************************************************
;* Microword Macros (Note: If ADDR not specified, 'ADDR'+1 is used).
;
;  Left word = ((value AND .Mxxxx) times .Lxxxx) ! next field ...
;  Right word = ((value AND .Mxxxx) times .Rxxxx) ! next field ...
;
;  Samples:      MWORD   <J=1000,OENA,SORC=2,DEST=3>
;                MWORD   <J=0,OENA,SORC=2,DEST=3>
;#*********************************************************************

; CALC - Insert field value into the microwords under construction

        DEFINE  CALC($ARG1)<
        IFE .L'$ARG1-1B11,<%ADDR==$ARG2>
        IFN .L'$ARG1-1B11,<%ML=%ML!<<$ARG2&.M'$ARG1>*.L'$ARG1>
                           %MR=%MR!<<$ARG2&.M'$ARG1>*.R'$ARG1>>>

; CONCAT - Build string argument

        DEFINE  CONCAT($A)<
        $B=<SIXBIT /$A/>_-36            ; get right justified character in
        $CHR=$CHR_6+$B>                 ;   octal and accumulate in string

; FIELD - Decode argument in the form XXXX=NNNN into XXXX and NNNN

        DEFINE  FIELD($ARG)<
        $CHR=0                          ; init string argument
        $GARG=0                         ; init numeric argument
        IRPC $ARG,<IFN $GARG,<$ARG2=$ARG2*8+$ARG>
                   IFIDN "=",$ARG,<$GARG=1
                                 $ARG2=0>
                   IFE $GARG,<CONCAT $ARG>
                   IFE $GARG,<$ARG2=1>>
        CALC \'$CHR>                    ; enter into microword

; MWORD - Build microwords

        DEFINE  MWORD ($FLD)<
        %ML==0                          ; init left word
        %MR==0                          ; init right word
        IRP $FLD,<DEFINE MFLD($A)<FIELD $A>
                  MFLD   $FLD>
        %ML==%ML!%ADDR_^D24
        %ADDR==%ADDR+1
        EXP     %ML,%MR>                ; done
```

```
                                 SUBTTL  *PARAM* CONSOLE DATA SWITCH ASSIGNMENTS, SEPT 18,1979

                                 DEFINE  S,<;  *********************************************************************>

                                 ;  *********************************************************************
                                 ;*DATA SWITCHES (READ FROM CONSOLE IN EXEC MODE OR TYPED IN IN USER MODE)
                                 ;*LEFT HALF SWITCHES ARE PRE-ASSIGNED FOR SUBROUTINE PACKAGE USE
                                 ;*AND CONTROL LOOPING, PRINTING (TTY OR OTHER DEVICE) AND MISC. FUNCTIONS
                                 ;  *********************************************************************

              400000             ABORT== 400000                  ;ABORT PROGRAM ON PASS COMPLETION
              200000             RSTART==200000                  ;RESTART TEST, PRINT TOTALS
              100000             TOTALS==100000                  ;PRINT TOTALS, CONTINUE

              040000             NOPNT== 040000                  ;INHIBIT ALL PRINT/TYPE OUT (EXCEPT FORCED)
              020000             PNTLPT==020000                  ;PRINT ALL DATA ON LPT (LOGICAL DEVICE, USER MODE)
              010000             DING==  010000                  ;RING BELL ON ERROR

              004000             LOOPER==004000                  ;ENTER EXERCISE/CHECK LOOP ON ERROR
              002000             ERSTOP==002000                  ;HALT ON TEST ERROR
              001000             PALERS==001000                  ;PRINT ALL ERRORS

              000400             RELIAB==000400                  ;RELIABILITY MODE
              000200             TXTINH==000200                  ;INHIBIT ERROR TEXT
              000100             INHPAG==000100                  ;INHIBIT PAGING

              000040             MODDVC==000040                  ;MODIFY DEVICE CODE
              000020             INHCSH==000020                  ;INHIBIT CACHE
              000010             OPRSEL==000010                  ;OPERATOR SELECTION

              000004             CHAIN== 000004                  ;CHAIN CONTROL SWITCH

              000002             KAHZ50==000002                  ;KA10 50 HERTZ POWER

                                                                 ;SWITCH 17 RESERVED !!!
```

```
                             SUBTTL  *PARAM* PROGRAM/SUBROUTINE PARAMETERS, SEPT 18,1979

                             ; ************************************************************************
                             ;*SPECIAL SUBPROGRAM LINKAGES
                             ; ************************************************************************

             027772          FSELNK= 27772   ;FILE SELECT LINK
             027773          FRDLNK= 27773   ;FILE READ LINK
             027774          LDLNK=  27774   ;LOAD LINKAGE ADDRESS
             027775          DDTLNK= 27775   ;DDT LINKAGE ADDRESS
             027776          MODLNK= 27776   ;OPERATIONAL MODE CHECK LINKAGE ADDRESS
             027777          SUBLNK= 27777   ;SUBROUTINE LINKAGE ADDRESS

                             ; ************************************************************************
                             ;*SPECIAL SUBROUTINE FATAL HALTS
                             ;*USED TO REPORT ERRORS THAT CAUSE THE SUBROUTINES TO BE UNUSABLE
                             ; ************************************************************************

                             ;ADDRESS   TAG  REASON
                             ;---------------------

                             ; 1010  NOEXEC  ;PROGRAM NOT CODED FOR EXEC MODE OPERATION
                             ; 1011  PLERR   ;FATAL PUSH LIST POINTER ERROR
                             ; 1012  PLERR1  ;INITIAL PUSH LIST POINTER ERROR
                             ; 1013  MUOERR  ;MUUO WITH LUUO HANDLER WIPED OUT
                             ; 1014  DTEBER  ;DTE20 INTERRUPT WITHOUT DOORBELL
                             ; 1015  DTECER  ;DTE20 CLOCK INTERRUPT WITHOUT FLAG SET
                             ; 1016  CPIERR  ;CPU INITIALIZATION ERROR
                             ; 1017  EOPERR  ;END OF PROGRAM ERROR
                             ; 1020  LUOERR  ;INTERRUPT WITH LUUO HANDLER WIPED OUT

                             ; ************************************************************************
```

```
                                ; *******************************************************************
                                ;OPERATOR DEFINITIONS (NON-UUO'S)
                                ; *******************************************************************

        260740  000000          OPDEF   GO      [PUSHJ  P,]     ;SUBROUTINE CALL
        263740  000000          OPDEF   RTN     [POPJ   P,]     ;SUBROUTINE RETURN
        261740  000000          OPDEF   PUT     [PUSH   P,]     ;PUT DATA ON PUSH LIST
        262740  000000          OPDEF   GET     [POP    P,]     ;GET DATA FROM PUSH LIST
        254000  000000          OPDEF   PJRST   [JRST   ]       ;JRST TO ROUTINE THAT RTN'S
        254200  000000          OPDEF   HALT    [JRST   4,]     ;DEFINITION FOR DDT
        254100  000000          OPDEF   JRSTF   [JRST   2,]     ;DEFINITION FOR DDT
        254500  000000          OPDEF   JEN     [JRST   12,]    ;DEFINITION FOR DDT

                                ; *******************************************************************
                                ;SUBROUTINE INITIALIZATION CALL
                                ; *******************************************************************

        265000  030011          OPDEF   PGMINT  [JSP    0,SBINIT]       ;SUBROUTINE INITIALIZATION

                                ; *******************************************************************
                                ;HALTING UUO'S (A MORE GRACEFUL HALT THAN SIMPLY USING THE HALT INSTRUCTION).
                                ; *******************************************************************

        037640  000004          OPDEF   FATAL   [37B8!15B12!4]  ;FATAL PROGRAMMING HALT
        037600  000004          OPDEF   ERRHLT  [37B8!14B12!4]  ;PROGRAM ERROR HALT

                                ; *******************************************************************
                                ;TERMINAL INPUT UUO'S
                                ;ALWAYS COME FROM THE CONSOLE TERMINAL IN EXEC MODE OR THE
                                ;CONTROLLING TERMINAL (REAL TERMINAL OR PTY) IN USER MODE.
                                ; *******************************************************************

        037000  000003          OPDEF   TTICHR  [37B8!0B12!3]   ;TTY, INPUT ANY CHARACTER
        037040  000003          OPDEF   TTIYES  [37B8!1B12!3]   ;TTY, NORMAL RETURN Y
        037100  000003          OPDEF   TTINO   [37B8!2B12!3]   ;TTY, NORMAL RETURN N
        037140  000003          OPDEF   TTIOCT  [37B8!3B12!3]   ;TTY, INPUT OCTAL WORD
        037200  000003          OPDEF   TTIDEC  [37B8!4B12!3]   ;TTY, INPUT DECIMAL WORD
        037240  000003          OPDEF   TTICNV  [37B8!5B12!3]   ;TTY, INPUT CONVERTABLE WORD
        037300  000003          OPDEF   TTLOOK  [37B8!6B12!3]   ;TTY, KEYBOARD CHECK
        037340  000003          OPDEF   TTALTM  [37B8!7B12!3]   ;TTY, ALT-MODE CHECK
        037400  000003          OPDEF   TTSIXB  [37B8!10B12!3]  ;TTY, INPUT SIXBIT WORD
        037440  000003          OPDEF   TTYINP  [37B8!11B12!3]  ;TTY, IMAGE MODE INPUT
        037500  000003          OPDEF   TTICLR  [37B8!12B12!3]  ;TTY, CLEAR INPUT
```

```
 1092                                   ;TERMINAL OUTPUT UUO'S.
 1093
 1094           037000  000000          OPDEF   PNTA    [37B8!0B12!0]   ;PRINT ASCII WORD
 1095           037000  000001          OPDEF   PNTAF   [37B8!0B12!1]   ;PRINT ASCII WORD FORCED
 1096           037740  000000          OPDEF   PNTAL   [37B8!17B12!0]  ;PRINT ASCIZ LINE
 1097           037740  000001          OPDEF   PNTALF  [37B8!17B12!1]  ;PRINT ASCIZ LINE FORCED
 1098           037600  000003          OPDEF   PSIXL   [37B8!14B12!3]  ;PRINT SIXBIT'Z LINE
 1099           037640  000003          OPDEF   PSIXLF  [37B8!15B12!3]  ;PRINT SIXBIT'Z LINE FORCED
 1100           037000  000000          OPDEF   PNTMSG  [37B8!0B12!0]   ;PRINT MESSAGE IMMEDIATE
 1101           037040  000000          OPDEF   PNTMSF  [37B8!1B12!0]   ;PRINT MESSAGE IMMEDIATE FORCED
 1102           037100  000000          OPDEF   PSIXM   [37B8!2B12!0]   ;PRINT SIXBIT'Z MSG IMMEDIATE
 1103           037200  000000          OPDEF   PSIXMF  [37B8!4B12!0]   ;PRINT SIXBIT'Z MSG IMM FORCED
 1104           037000  000000          OPDEF   PNTCI   [37B8!0B12!0]   ;PRINT CHARACTER IMMEDIATE
 1105           037040  000000          OPDEF   PNTCIF  [37B8!1B12!0]   ;PRINT CHARACTER IMMEDIATE FORCED
 1106           037500  000000          OPDEF   PNTCHR  [37B8!12B12!0]  ;PRINT CHARACTER
 1107           037500  000001          OPDEF   PNTCHF  [37B8!12B12!1]  ;PRINT CHARACTER FORCED
 1108           037040  000000          OPDEF   PNT1    [37B8!1B12!0]   ;PRINT ONE OCTAL DIGIT
 1109           037040  000001          OPDEF   PNT1F   [37B8!1B12!1]   ;PRINT 1 OCTAL DIGIT FORCED
 1110           037100  000000          OPDEF   PNT2    [37B8!2B12!0]   ;PRINT TWO OCTAL DIGITS
 1111           037100  000001          OPDEF   PNT2F   [37B8!2B12!1]   ;PRINT 2 OCTAL DIGITS FORCED
 1112           037140  000000          OPDEF   PNT3    [37B8!3B12!0]   ;PRINT THREE OCTAL DIGITS
 1113           037140  000001          OPDEF   PNT3F   [37B8!3B12!1]   ;PRINT THREE OCTAL DIGITS FORCED
 1114           037200  000000          OPDEF   PNT4    [37B8!4B12!0]   ;PRINT FOUR OCTAL DIGITS
 1115           037200  000001          OPDEF   PNT4F   [37B8!4B12!1]   ;PRINT FOUR OCTAL DIGITS FORCED
 1116           037240  000000          OPDEF   PNT5    [37B8!5B12!0]   ;PRINT FIVE OCTAL DIGITS
 1117           037240  000001          OPDEF   PNT5F   [37B8!5B12!1]   ;PRINT FIVE OCTAL DIGITS FORCED
 1118           037300  000000          OPDEF   PNT6    [37B8!6B12!0]   ;PRINT SIX OCTAL DIGITS
 1119           037300  000001          OPDEF   PNT6F   [37B8!6B12!1]   ;PRINT SIX OCTAL DIGITS FORCED
 1120           037340  000000          OPDEF   PNT7    [37B8!7B12!0]   ;PRINT 7 OCTAL DIGITS
 1121           037340  000001          OPDEF   PNT7F   [37B8!7B12!1]   ;PRINT 7 OCTAL DIGITS FORCED
 1122           037440  000000          OPDEF   PNT11   [37B8!11B12!0]  ;PRINT 11 OCTAL DIGITS
 1123           037440  000001          OPDEF   PNT11F  [37B8!11B12!1]  ;PRINT 11 OCTAL DIGITS FORCED.
 1124           037400  000000          OPDEF   PNTADR  [37B8!10B12!0]  ;PRINT PHYSICAL ADDRESS
 1125           037400  000001          OPDEF   PNTADF  [37B8!10B12!1]  ;PRINT PHYSICAL ADDRESS FORCED
 1126           037600  000000          OPDEF   PNTOCT  [37B8!14B12!0]  ;PRINT FULL WORD OCTAL
 1127           037600  000001          OPDEF   PNTOTF  [37B8!14B12!1]  ;PRINT FULL WORD OCTAL FORCED
 1128           037540  000000          OPDEF   PNTHW   [37B8!13B12!0]  ;PRINT OCTAL HALF WORDS, 6 SP 6
 1129           037540  000001          OPDEF   PNTHWF  [37B8!13B12!1]  ;PRINT OCTAL HALF WORDS, 6 SP 6 FORCED
 1130           037700  000003          OPDEF   PNTOCS  [37B8!16B12!3]  ;PRINT OCTAL, SUPPRESS LEADING 0'S
 1131           037740  000003          OPDEF   PNTOCF  [37B8!17B12!3]  ;PRINT OCTAL, SUPPRESS LEADING 0'S FORCED
 1132           037640  000000          OPDEF   PNTDEC  [37B8!15B12!0]  ;PRINT DECIMAL, SUPRESS LEADING 0'S
 1133           037640  000001          OPDEF   PNTDCF  [37B8!15B12!1]  ;PRINT DECIMAL, SUPRESS LEADING 0'S FORCED
 1134           037700  000000          OPDEF   PNTDS   [37B8!16B12!0]  ;PRINT DECIMAL, SPACES FOR LD 0'S
 1135           037700  000001          OPDEF   PNTDSF  [37B8!16B12!1]  ;PRINT DECIMAL, SPACES FOR LD 0'S FORCED
```

```
        037200  000002          OPDEF   PNTNM   [37B8!4B12!2]   ;PRINT PROGRAM NAME
        037000  000002          OPDEF   PNTSIX  [37B8!0B12!2]   ;PRINT SIXBIT WORD
        037040  000002          OPDEF   PNTSXF  [37B8!1B12!2]   ;PRINT SIXBIT WORD FORCED
        037240  000002          OPDEF   DROPDV  [37B8!5B12!2]   ;CLOSE LOGICAL FILE, USER MODE
        037100  000002          OPDEF   PNTCW   [37B8!2B12!2]   ;PRINT DF10 CONTROL WORD
        037140  000002          OPDEF   PNTCWF  [37B8!3B12!2]   ;PRINT DF10 CONTROL WORD FORCED
        037000  030242          OPDEF   PCRL    [37B8!0B12!CRLF] ;PRINT CARRIAGE RETURN/LINE FEED
        037040  030242          OPDEF   PCRLF   [37B8!1B12!CRLF] ;PRINT CARRIAGE RETURN/LINE FEED FORCED
        037000  000040          OPDEF   PSP     [37B8!0B12!40]  ;PRINT SPACE
        037040  000040          OPDEF   PSPF    [37B8!1B12!40]  ;PRINT SPACE FORCED
        037000  030243          OPDEF   PCRL2   [37B8!0B12!CRLF2] ;PRINT CARRIAGE RETURN/LINE FEED (TWICE)
        037040  030243          OPDEF   PCRL2F  [37B8!1B12!CRLF2] ;PRINT CARRIAGE RETURN/LINE FEED (TWICE) FORCED
        037040  000007          OPDEF   PBELL   [37B8!1B12!7]   ;PRINT TTY BELL

        037040  000026          OPDEF   PFORCE  [37B8!1B12!26]  ;PRINT FORCE, CONTROL O OVERRIDE

                                DEFINE  PMSG    (ARG),<
                                        PSIXM   [SIXBIT\ARG'_\]>

                                DEFINE  PMSGF   (ARG),<
                                        PSIXMF  [SIXBIT\ARG'_\]>

                                ;SIXBTZ -- MACRO TO GENERATE SIXBIT DATA FOR PRINTING
                                ;         CONSERVES CORE OVER ASCIZ

                                DEFINE  SIXBTZ  (ARG),< [SIXBIT\ARG'_\]>

                                ;CONSOLE SWITCH INPUT UUO.
                                ;READS CONSOLE SWITCHES IF IN EXEC MODE OR ASKS FOR THEM IF
                                ; USER MODE.

        037400  000002          OPDEF   SWITCH  [37B8!10B12!2]  ;INPUT CONSOLE SWITCHES

                                ;CLOCK INITIALIZATION UUO - TO SET DESIRED CLOCK OPERATION
                                ;EITHER IGNORE CLOCK, ONLY LET IT TICK OR CAUSE INTERRUPT TO OCCUR.

        037540  000004          OPDEF   CLOKOP  [37B8!13B12!4]  ;CLOCK OPERATION UUO - PDP-11 CLOCK
        037200  000004          OPDEF   MTROP   [37B8!4B12!4]   ;CLOCK OPERATION UUO - DK20 METER

                                ;KL10 ONLY CACHE OPERATION UUO'S

        037040  000004          OPDEF   CINVAL  [37B8!1B12!4]   ;CACHE INVALIDATE
        037100  000004          OPDEF   CFLUSH  [37B8!2B12!4]   ;CACHE FLUSH
        037140  000004          OPDEF   CWRTBI  [37B8!3B12!4]   ;CACHE WRITE-BACK & INVALIDATE
```

```
                                        ;END OF PASS/PROGRAM UUOS

                                        ;PERFORMS THE END OF PASS FUNCTIONS. INCREMENT PASS COUNT,
                                        ;DECREMENT ITERATION COUNT, CHECK IF FINISHED WITH THIS PROGRAM ETC.

                037500  000004          OPDEF   ENDUUO  [37B8!12B12!4]  ;UUO TO DISPLAY LIGHTS
                037700  000004          OPDEF   EOPUUO  [37B8!16B12!4]  ;END OF PROGRAM UUO

                                        ;MEMORY MANAGEMENT UUO'S
                                        ;UUO'S TO PERFORM VARIOUS MEMORY FUNCTIONS. MAPPING, ZEROING, PAGING,
                                        ;ADDRESS CONVERSION, ETC...

                037000  000004          OPDEF   MAPMEM  [37B8!0B12!4]   ;MAP MEMORY
                037500  000002          OPDEF   MEMZRO  [37B8!12B12!2]  ;ZERO MEMORY
                037440  000002          OPDEF   MEMSEG  [37B8!11B12!2]  ;SETUP MEMORY SEGMENT
                037540  000002          OPDEF   MAPADR  [37B8!13B12!2]  ;VIRTUAL TO PHYSICAL ADR CONVERT
                037640  000002          OPDEF   MAPCNK  [37B8!15B12!2]  ;MAP MEMORY CHUNK
                037600  000002          OPDEF   MAPSET  [37B8!14B12!2]  ;SET KI10 EXEC PAGE MAP
                037740  000002          OPDEF   MAPPNT  [37B8!17B12!2]  ;PRINT MEMORY MAP

                                        ;DEVICE CODE MODIFICATION UUO
                                        ;ALLOWS THE MODIFICATION OF IOT'S TO ONE DEVICE TO BE CHANGED TO
                                        ;IOT'S TO A DIFFERENT DEVICE CODE.

                037340  000002          OPDEF   MODPCU  [37B8!7B12!2]   ;MODIFY PERHIPERAL CODE, USER
                037300  000002          OPDEF   MODPCP  [37B8!6B12!2]   ;MODIFY PERHIPERAL CODE, PROGRAM

                        030000                  IFNDEF  MODDVL,<MODDVL==BEGIN>
                        030000                  IFNDEF  MODDVU,<MODDVU==BEGIN>

                                        ;"DIAMON" FILE SELECTION AND READ UUOS

                037240  000004          OPDEF   FSELECT [37B8!5B12!4]   ;FILE SELECTION
                037300  000004          OPDEF   FREAD   [37B8!6B12!4]   ;FILE READ - ASCII DATA
                037340  000004          OPDEF   FRD36   [37B8!7B12!4]   ;FILE READ - 36 BIT DATA
                037400  000004          OPDEF   FRD8    [37B8!10B12!4]  ;FILE READ - 8 BIT DATA

                                        ;KI10 ONLY UUO FOR PRINTING MARGIN VALUES

                037700  000002          OPDEF   PNTMGN  [37B8!16B12!2]  ;PRINT MARGIN VALUE

                                                XLIST
                                        IFNDEF  KLOLD,<LIST
```

```
                                SUBTTL  ERROR HANDLING UUO DEFINITIONS, SEPT 18,1979

                                ; ***********************************************************************
                                ;ERROR HANDLER PARAMETERS
                                ; ***********************************************************************

        036000  000000          OPDEF   ERUUO   [36B8]          ;ERROR CALL UUO
        035000  000000          OPDEF   ERLOOP  [35B8]          ;ERROR LOOP, CHECKS PC,REPT,REPT1,ERROR
        035040  000000          OPDEF   ERLP1   [35B8!1B12]     ;ERROR LOOP IF PC'S MATCH
        035100  000000          OPDEF   ERLP2   [35B8!2B12]     ;ERROR LOOP IF ANY ERROR
        034000  000000          OPDEF   REPTUO  [34B8]          ;REPEAT LOOP UUO

                                ;THE ERROR HANDLER MACROS

                                ;A MACRO TO REPORT AN ERROR AND LOOP

                                        DEFINE  ERROR   (ADR,FORMAT,CORECT,ACTUAL,F,D,ERR)<
                                                                                SALL
                                        ERUUO   FORMAT,[T,,[SIXBIT\F'_\]
                                                   CORECT,,ACTUAL
                                                   [SIXBIT\D'_\],,ERR]
                                                                                XALL
                                        ERLOOP  ADR             ;IF ERROR, LOOP TO ADR
                                >

                                ;A MACRO TO REPORT AN ERROR AND NOT LOOP

                                        DEFINE  ERROR1  (FORMAT,CORECT,ACTUAL,F,D,ERR)<
                                                                                SALL
                                        ERUUO   FORMAT,[T,,[SIXBIT\F'_\]
                                                   CORECT,,ACTUAL
                                                   [SIXBIT\D'_\],,ERR]
                                                                                XALL>

                                >;END OF KLOLD CONDITIONAL

                                        XLIST
                                        LIST
                                        SUBTTL  *FIXED* FIXED CONTROL AND DISPATCH STORAGE, SEPT 18,1979

030000                                          LOC     30000

                                        ; ***********************************************************************
                                        ;PROGRAM STARTING ADDRESSES
                                        ;THESE ADDRESSES CALL VARIOUS SPECIAL START ROUTINES AND OR OPTIONS
                                        ;NORMAL START ADDRESS IS 30000 ALL OTHERS ARE SPECIAL. INVOKED BECAUSE
                                        ;OF END OF PASS, POWER FAILURE, DDT START, RE-ENTERING(TYPICALLY USER
                                        ;MODE), OR ANY NUMBER OF SPECIAL FEATURE TESTS.
                                        ; ***********************************************************************

030000  254 00 1 00 027776              BEGIN:  JRST    @MODLNK         ;STAND-ALONE START
030001  254 00 0 00 000000*             $START: JRST    START           ;MODE CHECK STARTING ADDRESS

030002  254 00 1 00 027774              DIAGMN: JRST    @LDLNK          ;DIAGNOSTIC MONITOR START

030003  254 00 1 00 027774              SYSEXR: JRST    @LDLNK          ;SYSTEM EXERCISER START

030004  254 00 0 00 030000              SFSTRT: JRST    SADR1           ;SPECIAL FEATURE START

030005  254 00 0 00 030000              PFSTRT: JRST    SADR2           ;POWER FAIL RESTART

030006  254 00 0 00 030000              REENTR: JRST    SADR3           ;REENTER START(USUALLY USER MODE ONLY)

030007                                  SRTDDT:                         ;COMMONLY MISTAKEN NAME FOR "DDTSRT"
030007  254 00 1 00 027775              DDTSRT: JRST    @DDTLNK         ;DDT START

030010  254 00 0 00 000000*             BEGIN1: JRST    STARTA          ;LOOP START(END OF PASS COMES HERE)
030011  254 00 1 00 027777              SBINIT: JRST    @SUBLNK         ;PMGINT LINKAGE
030012  000000  000000                  RETURN: 0                       ;RETURN ADDRESS STORAGE

030013  254200  030001*                 START1: SADR7                   ;OPTIONAL STARTING ADR/INSTRUCTIONS
030014  254200  030013*                 START2: SADR8                   ; "
030015  254200  030014*                 START3: SADR9                   ; "
030016  254200  030015*                 START4: SADR10                  ; "
030017  254200  030016*                 START5: SADR11                  ; "
```

```
                                        ; **********************************************************************
                                        ;PROGRAM FIXED PARAMETER AREA
                                        ; **********************************************************************

030020  444660  644100                  PNTNAM: PAREA3          ;SIXBIT PROGRAM NAME
030021  605664  000000                  PNTEXT: PAREA4          ;SIXBIT PROGRAM EXTENSION
030022  711523  655207                  RANDBS: PAREA1          ;RANDOM BASE NUMBER
030023  000000  000000                  SWTEXR: PAREA2          ;SYSTEM EXERCISER SWITCHES
030024  000000  000005                  ITRCNT: ITERAT          ;PROGRAM ITERATIONS
030025  000000000000#                   $PNAME: PGMNAM          ;POINTER TO PROGRAMS NAME
030026  000000  000004                  $PVER:  MCNVER,,DECVER  ;MCN & DEC VERSION LEVEL
030027  000000  030000                  $MODVL: MODDVL          ;DEVICE CODE CHANGE LOWER LIMIT
030030  000000  030000                  $MODVU: MODDVU          ;DEVICE CODE CHANGE UPPER LIMIT
030031  777777  777777                  $EMODE: IFNDEF EXCASB,<0> IFDEF EXCASB,<-1>     ;EXEC ALLOWED
030032  777777  777777                  $UMODE: IFNDEF USRASB,<0> IFDEF USRASB,<-1>     ;USER ALLOWED
030033  000000  000000                  $DSKUP: IFNDEF DSKUPD,<0> IFDEF DSKUPD,<-1>     ;DISK UPDATE MODE
030034  777777  777777                  $MMAP:  IFNDEF MEMMAP,<0> IFDEF MEMMAP,<-1>     ;ALLOW MEMORY RTNS
030035  000000  000000                  PAREA7: PAREA5          ;OPTIONAL PARAMETER
030036  000000  000000                  PAREA8: PAREA6          ;OPTIONAL PARAMETER

                                        ; **********************************************************************
                                        ;PROGRAM VARIABLE PARAMETER AREA
                                        ; **********************************************************************

030037  000000  000000                  USER:   0               ; 0 = EXEC, -1 = USER MODE FLAG
030040  000000  000000                  KAIFLG: 0               ;PROCESSOR TYPE, 0 = KA10, -1 = KI10
030041  000000  000000                  KLFLG:  0               ;PROCESSOR TYPE, 0 = KA/KI, -1 = KL10
030042  777777  777777                  MONFLG: -1              ;DIAG MONITOR SPECIAL USER FLAG
030043  000000  000000                  MONCTL: 0               ;DIAG MON/SYS EXR FLAG
030044  000000  000000                  MONTEN: 0               ;-1= LOADED BY 10
030045  000000  000000                  CLOCKF: 0               ;CLOCK TICKED FLAG
030046  000000  000000                  CONSW:  0               ;CONSOLE SWITCH SETTINGS
030047  000000  000000                  PASCNT: 0               ;PROGRAM PASS COUNT
030050  000000  000000                  RUNFLG: 0               ;PROGRAM RUN FLAG
030051  000000  000000                  TESTPC: 0               ;SUBTEST PC
030052  000000  000000                  ERRPC:  0               ;ERROR PC
030053  000000  000000                  ERRTLS: 0               ;ERROR TOTALS
030054  000000  000000                  TICKS:  0               ;PROGRAM RUNNING TIME
030055  000000  000000                  MARGIN: 0               ;KI10 MARGIN WORD VALUE
030056  000000  000000                  $ONETM: 0               ;SUBROUTINE INITIALIZATION FLAG
```

```
                                        ; **************************************************************
                                        ;SPECIAL PROGRAM DISPATCH ADDRESSES
                                        ; **************************************************************

030057  037 12 0 00 000004              BEGEND: ENDUUO                  ;END OF PASS
030060  254 00 0 00 030010              $BEND1: JRST    BEGIN1          ;KEEP RUNNING PROGRAM
030061  037 16 0 00 000004              $BEND2: EOPUUO                  ;END OF PROGRAM - NO RETURN
030062  000000000000#                   CNTLC:  SADR5                   ;CONTROL C XFER ADDRESS
030063  000000  030007                  ALTMGO: SADR6                   ;ALTMODE XFER ADDRESS
030064                                  CPOPJ1:                         ;SKIP RETURN
030064  350 00 0 17 000000              UUOSKP: AOS     (P)             ;SKIP RETURN FROM UUO
030065                                  CPOPJ:                          ;NON-SKIP REGULAR RETURN
030065  263 17 0 00 000000              UUOEXT: RTN                     ;UUO RETURN
030066  255 00 0 00 000000              UUORTN: JFCL                    ;ADDITIONAL USERS UUO ROUTINE
030067  255 00 0 00 000000              $UORTX: JFCL                    ;ADDITIONAL UUO LINKAGE
030070  255 00 0 00 000000              $UUOER: JFCL                    ;INITED AS (JRST $UOERX)
030071  255 00 0 00 000000              $ITRHL: JFCL                    ;ADDITIONAL INTERRUPT LINKAGE
030072  255 00 0 00 000000              $ITRX1: JFCL                    ; "
030073  255 00 0 00 000000              $USRHL: JFCL                    ; "
030074  255 00 0 00 000000              $RSRTX: JFCL                    ;ADDITIONAL POWER FAIL LINKAGE
030075  255 00 0 00 000000              $RSRTY: JFCL                    ; "
030076  255 00 0 00 000000              RESRT1: JFCL                    ; INITED AS (JRST RESRTX)
030077  255 00 0 00 000000              RESRT2: JFCL                    ; "
030100  255 00 0 00 000000              $PARER: JFCL                    ;ADDITIONAL PARITY ERROR LINKAGE
030101  255 00 0 00 000000              ERMORE: JFCL                    ;ADDITIONAL ERROR HANDLER LINKAGE
030102  254 04 0 00 030102                      HALT    .               ;IMPROPER TRANSFER HALT

030103  000000  000000                  $PSHER: 0                       ;INITED AS (JRST PSHERR)
030104  000000  000000                  ITRCH1: 0                       ;PC & FLAGS OF CURRENT INTERRUPT
030105  000000  000000                          0                       ;INITED AS (JRST $ITRC1)

                                        ; **************************************************************
                                        ;PROCESSOR CONTROL STORAGE
                                        ; **************************************************************

030106  000000  000000                  $ACC0:  0                       ;INTERRUPT SAVED AC0
030107  000000  000000                  $SVPI:  0                       ;INTERRUPT SAVED PI
030110  000000  000000                  $SVAPR: 0                       ;INTERRUPT SAVED APR
030111  000000  000000                  $SVPAG: 0                       ;INTERRUPT SAVED PAG (DATAI)
030112  000000  000000                  $SPAG1: 0                       ;INTERRUPT SAVED PAG (CONI)

030113  000000  000000                  $SVUUO: 0                       ;CURRENT USERS UUO
030114  000000  000000                  $SVUPC: 0                       ;PC OF CURRENT USERS UUO

030115  000000  000000                  REPTU:  0                       ;REPEAT UUO ITERATIONS
030116  000000  000000                  SCOPE:  0                       ;ERROR HANDLER SCOPE LOOP FLAG
030117  000000  000000                  %CORFLG:0                       ; " CORRECT FLAG
030120  000000  000000                  %COREC: 0                       ; " CORRECT DATA
030121  000000  000000                  %ACTFL: 0                       ; " ACTUAL FLAG
030122  000000  000000                  %ACTUL: 0                       ; " ACTUAL DATA
030123  000000  000000                  %DISCR: 0                       ; " DISCREPENCY DATA
```

```
  1389                                  ; **********************************************************************
  1390                                  ;UUO DISPATCH TABLE
  1391                                  ; **********************************************************************
  1392                                          XLIST
  1393                                          LIST
  1394  030124  000000* 030070          UUODIS: LUU01,,$UUOER
  1395  030125  030070  000000*                 LUU03,,LUU02
  1396  030126  030070  030070                  LUU05,,LUU04
  1397  030127  000000* 000000*                 LUU07,,LUU06
  1398  030130  030070  030070                  LUU011,,LUU010
  1399  030131  030070  030070                  LUU013,,LUU012
  1400  030132  030070  030070                  LUU015,,LUU014
  1401  030133  030070  030070                  LUU017,,LUU016
  1402  030134  030070  030070                  LUU021,,LUU020
  1403  030135  030070  030070                  LUU023,,LUU022
  1404  030136  030070  030070                  LUU025,,LUU024
  1405  030137  000000* 030070                  LUU027,,LUU026
  1406  030140  030070  030070                  LUU031,,LUU030
  1407  030141  030070  030070                  LUU033,,LUU032
  1408
  1409                                  ; **********************************************************************
  1410                                  ;MEMORY MANAGMENT STORAGE
  1411                                  ; **********************************************************************
  1412
  1413  030142  000000  000000          DF22F:  0               ;DF10 CONTROL FLAG, 0 = 18, -1 = 22 BIT
  1414  030143  000000  000000          MAPNEW: 0               ;MEMORY MAPPING CONTROL FLAG, -1 = 4096K MAPPING
  1415  030144  000000  000000          MEMTOT: 0               ;TOTAL MEMORY SIZE IN K (1024.)
  1416  030145  000000  000000          MEMLOW: 0               ;LOWEST USABLE MEMORY
  1417  030146                          MEMSIZ: BLOCK ^D41      ;MEMORY SEGMENT POINTER TABLE
  1418
  1419                                  ; **********************************************************************
  1420                                  ;PRINT CONTROL STORAGE
  1421                                  ; **********************************************************************
  1422
  1423  030217  000000  000000          PNTFLG: 0               ;PRINT FLAG, -1 WHILE IN PRINT ROUTINE
  1424  030220  000000  000000          PNTENB: 0               ;PRINT ENABLE
  1425  030221  000000  000000          PDISF:  0               ;PRINT DISABLED FLAG
  1426  030222  000000  000000          PNTINH: 0               ;INHIBIT PRINT INPUT CHECKS
  1427  030223  000000  000000          PNTSPC: 0               ;PRINT SPACE CONTROL
  1428  030224  000000  000000          OPTIME: 0               ;TYPE-IN WAIT TIME
  1429  030225  000000  000000          $TWCNT: 0               ;TIME WAITED
  1430  030226  000000  000000          $DVOFF: 0               ;LOGICAL DEVICE INITED FLAG
  1431  030227  000000  000000          TTYFIL: 0               ;TTY EXEC FILLERS FLAG
  1432  030230  000000  000000          TTYSPD: 0               ;TTY EXEC BAUD RATE
  1433  030231  000000  000000          $TTCHR: 0               ;ACTUAL TYPED IN CHAR
  1434  030232  000000  000000          $CHRIN: 0               ;UPPER CASED & PARITY STRIPPED CHAR
  1435  030233  000000  000000          $TYPNB: 0               ;TYPED IN NUMBER
  1436  030234  000000  000000          $CRLF:  0               ;FREE CR/LF FLAG
  1437  030235  000000  000000          $TABF:  0               ;TAB CONVERSION FLAG
  1438  030236  000000  000000          $FFF:   0               ;FORM FEED CONVERSION FLAG
  1439  030237  000000  000000          $VTF:   0               ;VERTICAL TAB CONVERSION FLAG
  1440  030240  000000  000000          USRLFF: 0               ;USER LF FILLERS
  1441  030241  000000  000000          USRCRF: 0               ;USER CR FILLERS
```

```
                                          ; **********************************************************************
                                          ;THE FOLLOWING MISCELLANEOUS PRINT CHARACTERS ARE INCLUDED
                                          ;TO FACILITATE PRINTING AND ARE CALLED AS FOLLOWS:
                                          ;        MOVEI   NAME
                                          ;        PNTA            ;OR PNTAF
                                          ; **********************************************************************

030242                                    CRLF:   ASCII/
030242  015 012 000 000 000               /
030243                                    CRLF2:  ASCII/

030243  015 012 015 012 000               /
030244  054 000 000 000 000               COMMA:  ASCII/,/
030245  056 000 000 000 000               PERIOD: ASCII/./
030246  040 000 000 000 000               SPACE:  ASCII/ /
030247  011 000 000 000 000               TAB:    ASCII/  /
030250                                    MINUS:
030250  055 000 000 000 000               HYPEN:  ASCII/-/
030251  053 000 000 000 000               PLUS:   ASCII/+/
030252  052 000 000 000 000               AST:    ASCII/*/
030253  100 000 000 000 000               ATSIN:  ASCII/@/
030254  050 000 000 000 000               LFP:    ASCII/(/
030255  051 000 000 000 000               RTP:    ASCII/)/
030256  007 0000000000                    BELL:   BYTE (7) 007
030257  077 000 000 000 000               QUEST:  ASCII/?/
030260  057 000 000 000 000               SLASH:  ASCII!/!
030261  044 000 000 000 000               DOLLAR: ASCII/$/
030262  000000  000012                    RADIX:  ^D10                         ;DECIMAL PRINT RADIX
030263  000000  000040                    RADLSP: 40                           ;DECIMAL PRINT LEADING CHAR
030264  000000  000012                    RADLSC: ^D10                         ;DECIMAL PRINT LEADING CHAR COUNT

                                          ; **********************************************************************
                                          ;USER MODE OUTPUT FILE INFORMATION
                                          ; **********************************************************************

030265                                    $OBUF:  BLOCK   3                    ;LOGICAL FILE OUTPUT BUFFER HEADER
030270  60 62 51 56 64 00                 $OUTNM: SIXBIT  /PRINT/              ;FILE NAME
030271  60 56 64 00 00 00                 $OUTEX: SIXBIT  /PNT/                ;FILE NAME EXTENSION
030272                                            BLOCK   2

                                          ; **********************************************************************
                                          ;DISK UPDATE MODE FILE INFORMATION
                                          ; **********************************************************************

030274                                    $IBUF:  BLOCK   3
030277  60 62 51 56 64 00                 $INNM:  SIXBIT  /PRINT/
030300  60 56 64 00 00 00                 $INEXT: SIXBIT  /PNT/
030301                                            BLOCK   2
```

```
 1490                                    ; **********************************************************************
 1491                                    ;PUSHDOWN LIST CONTROL INFORMATION
 1492                                    ; **********************************************************************
 1493
 1494   030303  777577  030303           PLIST:  PLIST-PLISTE,,PLIST
 1495   030304                           PLISTS: BLOCK   200
 1496   030504  000000  000000           PLISTE: 0               ;END OF PUSHDOWN LIST
 1497
 1498                                    ; **********************************************************************
 1499                                    ;POWER LINE CLOCK FREQUENCY FLAG
 1500                                    ; **********************************************************************
 1501
 1502   030505  000000  000000           CYCL60: 0       ;0 = 60, -1 = 50 CYCLE
 1503
 1504                                    ; **********************************************************************
 1505                                    ;KL10 CACHE CONTROL FLAGS
 1506                                    ; **********************************************************************
 1507
 1508   030506  000000  000000           CSHFLG: 0       ;ALLOW CACHE IF 0
 1509   030507  000000  000000           CSHMEM: 0       ;CACHE MEMORY SEGMENTS IF 0
 1510
 1511                                    ; **********************************************************************
 1512                                    ;NUMBER INPUT DIGIT FLAG
 1513                                    ; **********************************************************************
 1514
 1515   030510  000000  000000           TTNBRF: 0       ;-1 IF ANY DIGIT TYPED
 1516
 1517                                    ; **********************************************************************
 1518                                    ;KL10 & KI10 "INHPAG" SWITCH PAGING PREVENTION
 1519                                    ; **********************************************************************
 1520
 1521   030511  000000  000000           PVPAGI: 0       ;IF NON-ZERO, OVERRIDE "INHPAG" SWITCH ACTION
 1522
 1523                                    ; **********************************************************************
 1524                                    ;ERROR REPORTING ROUTINE ADDITIONAL USERS CONTROL INSTRUCTIONS
 1525                                    ; **********************************************************************
 1526
 1527   030512  000000  000000           %ERHI1: 0       ;IF NON-ZERO, XCT'D AT START OF %ERUUO
 1528   030513  000000  000000           %ERHI2: 0       ;IF NON-ZERO, XCT'D AT END OF %ERUUO
 1529   030514  000000  000000           %ERHI3: 0       ;IF NON-ZERO, XCT'D AFTER "PC" OF %ERUUO
 1530
 1531                                    ; **********************************************************************
 1532                                    ;SPECIAL USERS UUO INTERCEPT INSTRUCTION
 1533                                    ; **********************************************************************
 1534
 1535   030515  000000  000000           $$UUO:  0       ;IF NON-ZERO, XCT'D AT START OF $UORTN
```

```
                                        ; ********************************************************************
                                        ;USER MODE MONITOR TYPE FLAG
                                        ; ********************************************************************

030516  000000  000000                  MONTYP: 0       ;0 = TOPS10, -1 = TOPS20

                                        ; ********************************************************************
                                        ;SPECIAL USERS MUUO INTERCEPT INSTRUCTION
                                        ; ********************************************************************

030517  000000  000000                  $$MUUO: 0       ;IF NON-ZERO, XCT'D AT START OF MUUOER

                                        ; ********************************************************************
                                        ;SPECIAL USERS USER MODE OUTPUT ERROR INTERCEPT INSTUCTION
                                        ; ********************************************************************

030520  000000  000000                  $$OUTER:0       ;IF NON-ZERO, XCT'D AT END OF USER MODE ERROR

                                        ; ********************************************************************
                                        ;''SWITCH'' CALL USAGE CONTROL
                                        ; ********************************************************************

030521  000000  000000                  $$TOGGLE:0      ;IF NON-ZERO, USE C(CONSW) FOR SWITCHES

                                        ; ********************************************************************
                                        ;SPECIAL USERS ALTMODE SWITCH CALL INTERCEPT INSTRUCTIONS
                                        ; ********************************************************************

030522  000000  000000                  $$TAX1: 0       ;IF NON-ZERO, XCT'D AT START OF ALTMODE SWITCH CALL
030523  000000  000000                  $$TAX2: 0       ;IF NON-ZERO, XCT'D AT END OF ALTMODE SWITCH CALL

                                        ; ********************************************************************
                                        ;SM10 (KS-10) PROCESSOR TYPE FLAG
                                        ; ********************************************************************

030524  000000  000000                  SM10:   0       ;IF -1 THIS IS A KS-10

                                        ; ********************************************************************
                                        ;RIGHT HALF SWITCHES PROMPT TABLE ADDRESS
                                        ; ********************************************************************

030525  000000  000000                  SWPTAB: 0       ;0 = NO PROMPT, ADR = ADR OF SIXBIT PROMPT TABLE

                                        ; ********************************************************************
                                        ;SPECIAL FUTURE EXPANSION ROOM
                                        ; ********************************************************************

                                        ; ********************************************************************
                                        ;END OF FIXED STORAGE
                                        ; ********************************************************************

030577                                          LOC     30577
030577  000000  000000                  ENDFIX: 0               ;END OF FIXED STORAGE
```

```
                SUBTTL  IPA REGISTER STORAGE

;#***********************************************************************
; End of DFPTAT,PARAM,FIXED,DFPTAE files
;#***********************************************************************

                XLIST

NO ERRORS DETECTED

PROGRAM BREAK IS 030601
ABSOLUTE BREAK IS 030600
CPU TIME USED 00:07.081

109P CORE USED
```

```
ABORT            400000  spd     CIPORT           000020  spd     E15              000017  spd     IDLE     000100  000000  spd
ADRPE    100000  000000  spd     CJRST            000011  spd     E16              000020  spd     IEXIT            000013  spd
ALL      770000  000000  spd     CLAST            000000  spd     E17              000021  spd     IJRST            000011  spd
ALTCHH           000000  ext     CLK              001000  spd     E18              000022  spd     ILAST            000000  spd
ALTCHK   007000  000000          CLKCLR           001000  spd     E19              000023  spd     INHCSH           000020  spd
ALTMGO           030063          CLKDIS           004000  spd     E2               000002  spd     INHFLT           200000  spd
ALU      020000  000000  spd     CLKENB           002000  spd     E20              000024  spd     INHMSG           100000  spd
ANXM             010000  spd     CLKU             001000  spd     E21              000025  spd     INHPAG           000100  spd
AROV             400000  spd     CLOCKF           030045          E22              000026  spd     INXM             000100  spd
AROVU            000010  spd     CLOKOP   037540  000004          E23              000027  spd     IOCLR            200000  spd
AST              030252          CMDQAV           000400  spd     E24              000030  spd     IPAER    000200  000000  spd
ATSIN            030253          CNTLC            030062          E3               000003  spd     ISETDE           000002  spd
BEGEND           030057          COMMA            030244          E4               000004  spd     ISETEI           000005  spd
BEGIN            030000          CONSW            030046          E5               000005  spd     ISETEX           000001  spd
BEGIN1           030010          CPOPJ            030065          E6               000006  spd     ISETID           000003  spd
BELL             030256          CPOPJ1           030064          E7               000007  spd     ISETIN           000004  spd
BIS              020000  spd     CRAMPE   004000  000000  spd     E8               000010  spd     ISTART           000012  spd
C1               000054  spd     CRLF             030242          E9               000011  spd     ITERAT           000005  spd
C10              000065  spd     CRLF2            030243          EBUS     400000  000000  spd     ITRCH1           030104
C11              000066  spd     CRY0             200000  spd     EBUSPE           004000  spd     ITRCNT           030024
C12              000067  spd     CRY1             100000  spd     ECOMP    000020  000000  spd     IWAIT            000010  spd
C13              000070  spd     CSETRD           000001  spd     ENDFIX           030577          JEN      254500  000000
C14              000071  spd     CSETRW           000003  spd     ENDUUO   037500  000004          JOB41            000041  spd
C15              000072  spd     CSETWR           000002  spd     EOPUUO   037700  000004          JOBAPR           000125  spd
C16              000073  spd     CSHFLG           030506          ERFLG            000015          JOBCNI           000126  spd
C17              000074  spd     CSHMEM           030507          ERLOOP   035000  000000          JOBDDT           000074  spd
C18              000075  spd     CSRCHN   100000  000000  spd     ERLP1    035040  000000          JOBFF            000121  spd
C19              000076  spd     CSRRQS   200000  000000  spd     ERLP2    035100  000000          JOBOPC           000130  spd
C2               000055  spd     CSTART           000004  spd     ERMORE           030101          JOBREL           000044  spd
C20              000077  spd     CWAIT            000007  spd     ERRHLT   037600  000004          JOBREN           000124  spd
C21              000100  spd     CWRTBI   037140  000004          ERRPC            030052          JOBSA            000120  spd
C22              000101  spd     CYCL60           030505          ERRTLS           030053          JOBSYM           000116  spd
C23              000102  spd     DCK              000040  spd     ERSTOP           002000  spd     JOBTPC           000127  spd
C24              000103  spd     DCOMP    000040  000000  spd     ERUUO    036000  000000          JOBUSY           000117  spd
C3               000056  spd     DDTLNK           027775          EXCASB           000001  spd     JOBUUO           000040  spd
C4               000057  spd     DDTSRT           030007          EXIOT            004000  spd     JOBVER           000137  spd
C5               000060  spd     DEBUG1           002000  spd     FATAL    037640  000004          JRSTF    254100  000000
C6               000061  spd     DEBUG2           001000  spd     FOV              040000  spd     JSYS     104000  000000
C7               000062  spd     DEBUG3           000400  spd     FOVU             000100  spd     KAHZ50           000002  spd
C8               000063  spd     DEBUG4           000200  spd     FQUERR           002000  spd     KAIFLG           030040
C9               000064  spd     DEBUG5           000100  spd     FRD36    037340  000004          KL10             000001  spd
CALL     200000  000000  spd     DECVER           000004  spd     FRD8     037400  000004          KLFLG            030041
CBUFSZ           000315  spd     DF22F            030142          FRDLNK           027773          LAPRAL           127520  spd
CBUS     100000  000000  spd     DIAGMN           030002          FREAD    037300  000004          LAPRP1           000001  spd
CCA              000014  spd     DING             010000  spd     FSELEC   037240  000004          LAPRP2           000002  spd
CCALL            000005  spd     DINIT    040000  000000  spd     FSELNK           027772          LAPRP3           000003  spd
CCALLC           000006  spd     DOLLAR           030261          FXU              000100  spd     LAPRP4           000004  spd
CCOMP            000010  spd     DROPDV   037240  000002          GENEPE           100000  spd     LAPRP5           000005  spd
CEXIT            000012  spd     DSPEAR           004000  spd     GET      262740  000000          LAPRP6           000006  spd
CFIBF    104000  000100          E1               000001  spd     GO       260740  000000          LAPRP7           000007  spd
CFLUSH   037100  000004          E10              000012  spd     GOTO     300000  000000  spd     LAROVT           000421  spd
CHAIN            000004  spd     E11              000013  spd     HALT     254200  000000          LAST     010000  000000  spd
CHNOFF           001000  spd     E12              000014  spd     HYPEN            030250          LAXER    000400  000000  spd
CHNON            002000  spd     E13              000015  spd     ICALL            000006  spd     LCASDE           000020  spd
CINVAL   037040  000004          E14              000016  spd     ICALLC           000007  spd     LCASLD           200000  spd
```

```
LCASLO          400000  spd   LOOPER          004000  spd   LUUO            000040  spd   MARGIN          030055
LCASWB          200000  spd   LOOPGM          040000  spd   LUUO1           000000* spd   MBCN            000016
LCASWD          000020  spd   LOOPTS          020000  spd   LUUO10          030070        MBERR    002000 000000  spd
LCCASD          020020  spd   LPAREN          001000  spd   LUUO11          030070        MBUS     010000 000000  spd
LCHNOF          001000  spd   LPARER          001000  spd   LUUO12          030070        MCBUS           000001  spd
LCHNON          002000        LPDOVT          000422  spd   LUUO13          030070        MCNVER          000000  spd
LCIOPF          020400  spd   LPFWPC          000426  spd   LUUO14          030070        MEMLOW          030145
LCNTRP          000434  spd   LPGFTR          000427  spd   LUUO15          030070        MEMMAP          000001  spd
LCNXER          022000  spd   LPICH1          000100  spd   LUUO16          030070        MEMSEG   037440 000002
LCPAER          021000  spd   LPICH2          000040  spd   LUUO17          030070        MEMSIZ          030146
LCPWRF          020100  spd   LPICH3          000020  spd   LUUO2           000000* spd   MEMTOT          030144
LCSBER          024000  spd   LPICH4          000010  spd   LUUO20          030070        MEMZRO   037500 000002
LCSLOA          200000  spd   LPICH5          000004  spd   LUUO21          030070        MINUS           030250
LCSLOO          400000  spd   LPICH6          000002  spd   LUUO22          030070        MLAST    400000 000000  spd
LCTRP           000435  spd   LPICH7          000001  spd   LUUO23          030070        MMPROC          000002  spd
LCWSX           000040  spd   LPICHA          000177  spd   LUUO24          030070        MODDVC          000040  spd
LDATAF          000200  spd   LPICLR          010000  spd   LUUO25          030070        MODDVL          030000  spd
LDCASD          040020  spd   LPIIP1          040000  spd   LUUO26          030070        MODDVU          030000  spd
LDIOPF          040400  spd   LPIIP2          020000  spd   LUUO27          000000*       MODLNK          027776
LDLNK           027774        LPIIP3          010000  spd   LUUO3           030070        MODPCP   037300 000002
LDNXER          042000  spd   LPIIP4          004000  spd   LUUO30          030070        MODPCU   037340 000002
LDPAER          041000  spd   LPIIP5          002000  spd   LUUO31          030070        MONCTL          030043
LDPWRF          040100  spd   LPIIP6          001000  spd   LUUO32          030070        MONFLG          030042
LDSBER          044000  spd   LPIIP7          000400  spd   LUUO33          030070        MONTEN          030044
LECASD          100020  spd   LPIOFF          000400  spd   LUUO4           030070        MONTYP          030516
LEIOPF          100400  spd   LPION           000200  spd   LUUO5           030070        MPER     200000 000000  spd
LENXER          102000  spd   LPNTRP          000436  spd   LUUO6           000000* spd   MPROC    200000 000000  spd
LEPAER          101000  spd   LPRCH1          000100  spd   LUUO7           000000* spd   MPRUN           000010  spd
LEPWRF          100100  spd   LPRCH2          000040  spd   LUUOI           000041  spd   MPVU            020000  spd
LESBER          104000  spd   LPRCH3          000020  spd   LWRITE          000100  spd   MSG      100000 000000  spd
LEUPFW          000420  spd   LPRCH4          000010  spd   M1              000031  spd   MTROP    037200 000004
LEVNCD          100000  spd   LPRCH5          000004  spd   M10             000042  spd   MVRERR          001000  spd
LEVNPA          400000  spd   LPRCH6          000002  spd   M11             000043  spd   NDCB     000200 000000  spd
LEVNPD          200000  spd   LPRCH7          000001  spd   M12             000044  spd   NDMP     000400 000000  spd
LEXCMP          000000  spd   LPTRP           000437  spd   M13             000045  spd   NEXM     020000 000000  spd
LFLGCL          020000  spd   LPWRFE          000100  spd   M14             000046  spd   NIPORT          000040  spd
LFLGDS          040000  spd   LPWRFL          000100  spd   M15             000047  spd   NOPNT           040000  spd
LFLGEN          100000  spd   LREQSE          004000  spd   M16             000050  spd   NOTWCO   040000 000000  spd
LFLGST          010000  spd   LRQCLR          020000  spd   M17             000051  spd   NXMU            010000  spd
LFP             030254        LSBSEN          004000  spd   M18             000052  spd   OPRSEL          000010  spd
LGWC     000100 000000  spd   LSBUSE          004000  spd   M19             000053  spd   OPTIME          030224
LINSTF          000400  spd   LSCASD          010020  spd   M2              000032  spd   OVN      000020 000000  spd
LINT            000010  spd   LSECMO          040000  spd   M3              000033  spd   P               000017
LIOCLR          200000  spd   LSIOPF          010400  spd   M4              000034  spd   PAG             000010  spd
LIOPFE          000400  spd   LSMODE          040000  spd   M5              000035  spd   PALERS          001000  spd
LKNTRP          000430  spd   LSNTRP          000432  spd   M6              000036  spd   PARCLR          200000  spd
LKTRP           000431  spd   LSNXER          012000  spd   M7              000037  spd   PARDIS          100000  spd
LLACBL          400000  spd   LSPAER          011000  spd   M8              000040  spd   PAREA1   711523 655207  spd
LLDUSB          100000  spd   LSPWRF          010100  spd   M9              000041  spd   PAREA2          000000  spd
LLPRCN          200000  spd   LSSBER          014000  spd   MAPADR   037540 000002        PAREA3   444660 644100  spd
LMUUO           000424  spd   LSTRP           000433  spd   MAPCNK   037640 000002        PAREA4   605664 000000  spd
LMUUOP          000425  spd   LTRP3T          000423  spd   MAPMEM   037000 000004        PAREA5          000000  spd
LNXMEN          002000  spd   LTRPAE          020000  spd   MAPNEW          030143        PAREA6          000000  spd
LNXMER          002000  spd   LTRPEN          020000  spd   MAPPNT   037740 000002        PAREA7          030035
LOGO     400000 000000  spd   LUSCMP          000040  spd   MAPSET   037600 000002        PAREA8          030036
```

```
DFPTA   MACRO %53B(1242) 07:46  1-Jul-85 Page S-3                                                      SEQ 0167
DFPTAE  MAC     20-Sep-82 09:16           SYMBOL TABLE

PARENB         040000  spd    PNT7F   037340 000001         RELIAB         000400  spd    SUBLNK         027777
PARU           004000  spd    PNTA    037000 000000         REPT           000015  spd    SWITCH  037400 000002
PASCNT         030047         PNTADF  037400 000001         REPT1          000016  spd    SWPTAB         030525
PAT            000014         PNTADR  037400 000000         REPTU          030115         SWTEXR         030023
PBELL   037040 000007         PNTAF   037000 000001         REPTUO  034000 000000         SYSEXR         030003
PCLEAR         400000  spd    PNTAL   037740 000000         REQSET         004000  spd    TAB            030247
PCRL    037000 030242         PNTALF  037740 000001         RESQAV         000200  spd    TCALL          000003  spd
PCRL2   037000 030243         PNTCHF  037500 000001         RESRT1         030076         TCALLC         000004  spd
PCRL2F  037040 030243         PNTCHR  037500 000000         RESRT2         030077         TCHECK         000005  spd
PCRLF   037040 030242         PNTCI   037000 000000         RETURN         030012         TCONT          000002  spd
PDEC    006000 000000         PNTCIF  037040 000000         RQINT   010000 000000  spd    TDENA   000100 000000  spd
PDECF   002000 000000         PNTCW   037100 000002         RSTART         200000  spd    TENLEN         000003  spd
PDISF          030221         PNTCWF  037140 000002         RTN     263740 000000         TESTPC         030051
PDLOVU         200000  spd    PNTDCF  037640 000001         RTP            030255         TEXIT          000007  spd
PDSABL         000040  spd    PNTDEC  037640 000000         RUNALL         010000  spd    TICKS          030054
PENABL         000020  spd    PNTDS   037700 000000         RUNFLG         030050         TJRST          000006  spd
PERIOD         030245         PNTDSF  037700 000001         SADR1          030000         TLAST          000000  spd
PFORCE  037040 000026         PNTENB         030220         SADR10  254200 000000*        TOTALS         100000  spd
PFSTRT         030005         PNTEXT         030021         SADR11  254200 000000*        TRACET         400000  spd
PGMINT  265000 030011         PNTFLG         030217         SADR2          030000         TSSTAR         000010  spd
PGMNAM         000000  ext    PNTHW   037540 000000         SADR3          030000         TSTART         000001  spd
PIA00          000004  spd    PNTHWF  037540 000001         SADR4          030007         TSTEBF         200000  spd
PIA01          000002  spd    PNTINH         030222         SADR5          000000*        TTALTM  037340 000003
PIA02          000001  spd    PNTLPT         020000  spd    SADR6          030007         TTICHR  037000 000003
PICHN1         000100  spd    PNTMGN  037700 000002         SADR7   254200 000000*        TTICLR  037500 000003
PICHN2         000040  spd    PNTMSF  037040 000000         SADR8   254200 000000*        TTICNV  037240 000003
PICHN3         000020  spd    PNTMSG  037000 000000         SADR9   254200 000000*        TTIDEC  037200 000003
PICHN4         000010  spd    PNTNAM         030020         SBINIT         030011         TTINO   037100 000003
PICHN5         000004  spd    PNTNM   037200 000002         SCOPE          030116         TTIOCT  037140 000003
PICHN6         000002  spd    PNTOCC  001000 000000         SCOPE1         000000  ext    TTIYES  037040 000003
PICHN7         000001  spd    PNTOCF  037740 000003         SCOPER  027000 000000         TTLOOK  037300 000003
PICHNA         000177  spd    PNTOCS  037700 000003         SELLAR         040000  spd    TTNBRF         030510
PICLR          010000  spd    PNTOCT  037600 000000         SEQ     040000 000000  spd    TTSIXB  037400 000003
PIOFF          000400  spd    PNTOTF  037600 000001         SFSTRT         030004         TTYFIL         030227
PION           000200  spd    PNTSIX  037000 000002         SHWC    000040 000000  spd    TTYINP  037440 000003
PJRST   254000 000000         PNTSPC         030223         SINCYC         020000  spd    TTYSPD         030230
PLIST          030303         PNTSXF  037040 000002         SLASH          030260         TUSER   001000 000000  spd
PLISTE         030504         PORTP   400000 000000  spd    SM10           030524         TXALL   060000 000000  spd
PLISTS         030304         PPDEC          030127  ext    SPACE          030246         TXNOT   040000 000000  spd
PLUS           030251         PPDECF         030125  ext    SPARE1         010000  spd    TXTINH         000200  spd
PNT1    037040 000000         PSIXL   037600 000003         SPARE2         000100  spd    TXYES   020000 000000  spd
PNT11   037440 000000         PSIXLF  037640 000003         SRTDDT         030007         USER           030037
PNT11F  037440 000001         PSIXM   037100 000000         SSCALL         000003  spd    USERF          010000  spd
PNT1F   037040 000001         PSIXMF  037200 000000         SSCHK          000004  spd    USRASB         000001  spd
PNT2    037100 000000         PSP     037000 000040         SSCONT         000002  spd    USRCRF         030241
PNT2F   037100 000001         PSPF    037040 000040         SSJRST         000005  spd    USRLFF         030240
PNT3    037140 000000         PUT     261740 000000         SSLAST         000000  spd    UUODIS         030124
PNT3F   037140 000001         PVPAGI         030511         SSSTRT         000001  spd    UUOEXT         030065
PNT4    037200 000000         PWFCLR         400000  spd    START          030017  ext    UUORTN         030066
PNT4F   037200 000001         QUEST          030257         START1         030013         UUOSKP         030064
PNT5    037240 000000         RADIX          030262         START2         030014         ZALU    000002 000000  spd
PNT5F   037240 000001         RADLSC         030264         START3         030015         ZCBUS   000004 000000  spd
PNT6    037300 000000         RADLSP         030263         START4         030016         ZEBUS          000000  spd
PNT6F   037300 000001         RANDBS         030022         START5         030017         ZMPROC  000003 000000  spd
PNT7    037340 000000         REENTR         030006         STARTA         030010  ext    ZSEQ    000001 000000  spd
```

```
DFPTA   MACRO %53B(1242) 07:46  1-Jul-85 Page S-4                                                      SEQ 0168
```

```
55

DFPTA     MACRO %53B(1242) 07:46  1-Jul-85 Page S-4                                                              SEQ 0168
DFPTAE    MAC       20-Sep-82 09:16           SYMBOL TABLE

$$MUUO        030517        %CORFL        030117        .LJ           010000  spd   .MCCCC        000037  spd
$$OUTE        030520        %DISCR        030123        .LJMAP        000000  spd   .MCCCE        000037  spd
$$TAX1        030522        %ERHI1        030512        .LJRP         000000  spd   .MCCCP        000037  spd
$$TAX2        030523        %ERHI2        030513        .LJSRP        000000  spd   .MCCEB        000037  spd
$$TOGG        030521        %ERHI3        030514        .LJZ          000000  spd   .MCCER        000037  spd
$$UUO         030515        .JB41         000041  spd   .LLDCT        000000  spd   .MCCFZ        000037  spd
$ACCO         030106        .JBAPR        000125  spd   .LLDLM        000000  spd   .MCCGC        000037  spd
$BEND1        030060        .JBCNI        000126  spd   .LLOOP        000000  spd   .MCCIA        000037  spd
$BEND2        030061        .JBDDT        000074  spd   .LLSAD        000000  spd   .MCCLW        000037  spd
$CHRIN        030232        .JBFF         000121  spd   .LMARK        000000  spd   .MCCMB        000037  spd
$CRLF         030234        .JBOPC        000130  spd   .LMGC         000001  spd   .MCCMP        000037  spd
$DEVCH        000017  spd   .JBREL        000044  spd   .LNAND        000000  spd   .MCCOF        000037  spd
$DSKUP        030033        .JBREN        000124  spd   .LOENA        002000  spd   .MCCON        000037  spd
$DVCH1        000016  spd   .JBSA         000120  spd   .LOP          000000  spd   .MCCPP        000037  spd
$DVOFF        030226        .JBSYM        000116  spd   .LOR          000000  spd   .MCCXA        000037  spd
$EMODE        030031        .JBTPC        000127  spd   .LPAR         004000  spd   .MCENA        000001  spd
$FFF          030236        .JBUSY        000117  spd   .LPLUS        000000  spd   .MCJP         000017  spd
$IBUF         030274        .JBUUO        000040  spd   .LPUSH        000000  spd   .MCJPP        000017  spd
$INEXT        030300        .JBVER        000137  spd   .LRAM         000000  spd   .MCJS         000017  spd
$INNM         030277        .LA           000000  spd   .LRDLM        000000  spd   .MCJV         000017  spd
$ITRHL        030071        .LADDR 000100 000000  spd   .LRFCT        000000  spd   .MCONT        000017  spd
$ITRX1        030072        .LAND         000000  spd   .LRMIN        000000  spd   .MCRTN        000017  spd
$LPAPE        000001  spd   .LB           000000  spd   .LRPCT        000000  spd   .MCRY         000001  spd
$MMAP         030034        .LBAD         000000  spd   .LSOA         000000  spd   .MD           000007  spd
$MODVL        030027        .LBUS         000000  spd   .LSOB         000000  spd   .MDISA        000001  spd
$MODVU        030030        .LCCAF        000000  spd   .LSOQ         000000  spd   .MENA         000001  spd
$OBUF         030265        .LCCBF        000000  spd   .LSAB         000000  spd   .MFUNC        000007  spd
$ONETM        030056        .LCCCA        000000  spd   .LSAQ         000000  spd   .MJ           007777  spd
$OUTEX        030271        .LCCCC        000000  spd   .LSDO         000000  spd   .MJMAP        000017  spd
$OUTNM        030270        .LCCCE        000000  spd   .LSDA         000000  spd   .MJRP         000017  spd
$PARER        030100        .LCCCP        000000  spd   .LSDQ         000000  spd   .MJSRP        000017  spd
$PNAME        030025        .LCCEB        000000  spd   .LSELC        000000  spd   .MJZ          000017  spd
$PSHER        030103        .LCCER        000000  spd   .LSELE        000000  spd   .MLDCT        000017  spd
$PVER         030026        .LCCFZ        000000  spd   .LSELF        000000  spd   .MLDLM        000037  spd
$RSRTX        030074        .LCCGC        000000  spd   .LSELM        000000  spd   .MLOOP        000017  spd
$RSRTY        030075        .LCCIA        000000  spd   .LSELP        000000  spd   .MLSAD        000037  spd
$SPAG1        030112        .LCCLW        000000  spd   .LSK          000000  spd   .MMARK        000001  spd
$START        030001        .LCCMB        000000  spd   .LSKCN        000000  spd   .MMGC         001777  spd
$SVAPR        030110        .LCCMP        000000  spd   .LSKMB        000000  spd   .MNAND        000007  spd
$SVPAG        030111        .LCCOF        000000  spd   .LSMIN        000000  spd   .MOENA        000001  spd
$SVPI         030107        .LCCON        000000  spd   .LSORC        000000  spd   .MOP          000017  spd
$SVUPC        030114        .LCCPP        000000  spd   .LSPAR        000000  spd   .MOR          000007  spd
$SVUUO        030113        .LCCXA        000000  spd   .LTIM         000000  spd   .MPAR         000001  spd
$TABF         030235        .LCENA        000000  spd   .LTWB         000000  spd   .MPLUS        000007  spd
$TTCHR        030231        .LCJP         000000  spd   .LXNOR        000000  spd   .MPUSH        000017  spd
$TWCNT        030225        .LCJPP        000000  spd   .LXOR         000000  spd   .MRAM         000001  spd
[illegible]NB 030233        .LCJS         000000  spd   .MA           000017  spd   .MRDLM        000037  spd
[illegible]DE 030032        .LCJV         000000  spd   .MADDR        007777  spd   .MRFCT        000017  spd
$UURTX        030067        .LCONT        000000  spd   .MAND         000007  spd   .MRMIN        000007  spd
$USRHL        030073        .LCRTN        000000  spd   .MB           000017  spd   .MRPCT        000017  spd
$UUOER        030070        .LCRY         000000  spd   .MBAD         000001  spd   .MSOA         000007  spd
$VTF          030237        .LD           000000  spd   .MBUS         000007  spd   .MSOB         000007  spd
%ACTFL        030121        .LDISA        000000  spd   .MCCAF        000037  spd   .MSOQ         000007  spd
%ACTUL        030122        .LENA         000000  spd   .MCCBF        000037  spd   .MSAB         000007  spd
%COREC        030120        .LFUNC        000000  spd   .MCCCA        000037  spd   .MSAQ         000007  spd

56

DFPTA     MACRO %53B(1242) 07:46  1-Jul-85 Page S-5
```

```
.MSDO              000007  spd     .RJ                000000  spd
.MSDA              000007  spd     .RJMAP             000040  spd
.MSDQ              000007  spd     .RJRP              000160  spd
.MSELC             000007  spd     .RJSRP             000120  spd
.MSELE             000007  spd     .RJZ               000000  spd
.MSELF             000007  spd     .RLDCT             000300  spd
.MSELM             000007  spd     .RLDLM             230000  spd
.MSELP             000007  spd     .RLOOP             000320  spd
.MSK               000037  spd     .RLSAD             200000  spd
.MSKCN             000037  spd     .RMARK             000002  spd
.MSKMB             000037  spd     .RMGC              000000  spd
.MSMIN             000007  spd     .RNAND    050000   000000  spd
.MSORC             000007  spd     .ROENA             000000  spd
.MSPAR             000001  spd     .ROP               000020  spd
.MTIM              000001  spd     .ROR      030000   000000  spd
.MTWB              000017  spd     .RPAR              000000  spd
.MXNOR             000007  spd     .RPLUS             000000  spd
.MXOR              000007  spd     .RPUSH             000100  spd
.PNTOC             000000  ext     .RRAM     000200   000000  spd
.RA       000010   000000  spd     .RRDLM             220000  spd
.RADDR             000000  spd     .RRFCT             000200  spd
.RAND     040000   000000  spd     .RRMIN    020000   000000  spd
.RB                400000  spd     .RRPCT             000220  spd
.RBAD              000001  spd     .RSOA     400000   000000  spd
.RBUS              001000  spd     .RSOB     300000   000000  spd
.RCCAF             050000  spd     .RSOQ     200000   000000  spd
.RCCBF             060000  spd     .RSAB     100000   000000  spd
.RCCCA             000000  spd     .RSAQ              000000  spd
.RCCCC             030000  spd     .RSDO     700000   000000  spd
.RCCCE             160000  spd     .RSDA     500000   000000  spd
.RCCCP             140000  spd     .RSDQ     600000   000000  spd
.RCCEB             040000  spd     .RSELC             004000  spd
.RCCER             100000  spd     .RSELE             005000  spd
.RCCFZ             020000  spd     .RSELF             003000  spd
.RCCGC             010000  spd     .RSELM             002000  spd
.RCCIA             110000  spd     .RSELP             001000  spd
.RCCLW             170000  spd     .RSK               010000  spd
.RCCMB             120000  spd     .RSKCN             240000  spd
.RCCMP             130000  spd     .RSKMB             210000  spd
.RCCOF             010000  spd     .RSMIN    010000   000000  spd
.RCCON             030000  spd     .RSORC    100000   000000  spd
.RCCPP             150000  spd     .RSPAR             000004  spd
.RCCXA             070000  spd     .RTIM              000010  spd
.RCENA    000400   000000  spd     .RTWB              000360  spd
.RCJP              000060  spd     .RXNOR    070000   000000  spd
.RCJPP             000260  spd     .RXOR     060000   000000  spd
.RCJS              000020  spd
.RCJV              000140  spd
.RCONT             000340  spd
.RCRTN             000240  spd
.RCRY              000400  spd
.RD       001000   000000  spd
.RDISA             000000  spd
.RENA     000400   000000  spd
.RFUNC    010000   000000  spd
```

```
ABORT      992#
ADRPE      103#
ALL        395#
ALTCMH       9#     457
ALTMGO    1346#
ALU        393#
ANXM      1260#
AROV      1260#
AROVU     1260#
AST       1461#
ATSIN     1462#
BEGEND    1342#
BEGIN       78       79       80     1207     1208     1273#
BEGIN1    1289#    1343
BELL      1465#
BIS       1260#
C1         265#
C10        274#
C11        275#
C12        276#
C13        277#
C14        278#
C15        279#
C16        280#
C17        281#
C18        282#
C19        283#
C2         266#
C20        284#
C21        285#
C22        286#
C23        287#
C24        288#
C3         267#
C4         268#
C5         269#
C6         270#
C7         271#
C8         272#
C9         273#
CALL       466#
CBUFSZ      63#
CBUS       391#
CCA       1260#
CCALL      512#
CCALLC     513#
CCOMP      515#
CEXIT      517#
CHAIN     1012#
CHNOFF    1260#
CHNON     1260#
CIPORT     130#
CJRST      516#
```

```
CLAST      507#
CLAST      507#
CLK       1260#
CLKCLR    1260#
CLKDIS    1260#
CLKENB    1260#
CLKU      1260#
CLOCKF    1328#
CMDQAV     431#
CNTLC     1345#
COMMA     1454#
CONSW     1329#
CPOPJ     1349#
CPOPJ1    1347#
CRAMPE     417#
CRLF      1142      1143      1449#
CRLF2     1146      1147      1451#
CRY0      1260#
CRY1      1260#
CSETRD     508#
CSETRW     510#
CSETWR     509#
CSHFLG    1508#
CSHMEM    1509#
CSRCHN     414#
CSRRQS     413#
CSTART     511#
CWAIT      514#
CYCL60    1502#
DCK       1260#
DCOMP      420#
DDTLNK    1026#     1287
DDTSRT    1287#
DEBUG1     124#
DEBUG2     125#
DEBUG3     126#
DEBUG4     127#
DEBUG5     128#
DECVER      25#     1308
DF22F     1413#
DIAGMN    1276#
DING       998#
DINIT      415#
DOLLAR    1468#
DSKUPD    1313
DSPEAR     123#
E1         220#
E10        229#
E11        230#
E12        231#
E13        232#
E14        233#
E15        234#
E16        235#
```

```
E17          236#
E18          237#
E19          238#
E2           221#
E20          239#
E21          240#
E22          241#
E23          242#
E24          243#
E3           222#
E4           223#
E5           224#
E6           225#
E7           226#
E8           227#
E9           228#
EBUS         389#
EBUSPE       428#
ECOMP        421#
ENDFIX      1588#
ERFLG         57#
ERMORE      1362#
ERRPC       1333#
ERRTLS      1334#
ERSTOP      1001#
EXCASB        72#     1311
EXIOT       1260#
FOV         1260#
FOVU        1260#
FQUERR       429#
FRDLNK      1024#
FSELNK      1023#
FXU         1260#
GENEPE       424#
GOTO         467#
HYPEN       1459#
ICALL        530#
ICALLC       531#
IDLE         419#
IEXIT        535#
IJRST        533#
ILAST        524#
INHCSH      1009#
INHFLT       118#
INHMSG       119#
INHPAG      1006#
INXM        1260#
IOCLR       1260#
IPAER        107#
ISETDE       526#
ISETEI       529#
ISETEX       525#
ISETID       527#
```

```
ISETIN       528#
ISETIN    528#
ISTART    534#
ITERAT     70#    1306
ITRCH1   1366#
ITRCNT   1306#
IWAIT     532#
JOB41    1260#
JOBAPR   1260#
JOBCNI   1260#
JOBDDT   1260#
JOBFF    1260#
JOBOPC   1260#
JOBREL   1260#
JOBREN   1260#
JOBSA    1260#
JOBSYM   1260#
JOBTPC   1260#
JOBUSY   1260#
JOBUUO   1260#
JOBVER   1260#
KA10     1260
KAHZ50   1014#
KAIFLG   1323#
KI10     1260
KL10       71#    1260
KL10P0   1260
KLFLG    1324#
KLOLD    1222
LAPRAL   1260#
LAPRP1   1260#
LAPRP2   1260#
LAPRP3   1260#
LAPRP4   1260#
LAPRP5   1260#
LAPRP6   1260#
LAPRP7   1260#
LAROVT   1260#
LAST      471#
LAXER     106#
LCASDE   1260#
LCASLD   1260#
LCASLO   1260#
LCASWB   1260#
LCASWD   1260#
LCCASD   1260#
LCHNOF   1260#
LCHNON   1260#
LCIOPF   1260#
LCNTRP   1260#
LCNXER   1260#
LCPAER   1260#
LCPWRF   1260#
LCSBER   1260#
```

```
LCSLOA    1260#
LCSLOO    1260#
LCTRP     1260#
LCWSX     1260#
LDATAF    1260#
LDCASD    1260#
LDIOPF    1260#
LDLNK     1025#    1276     1278
LDNXER    1260#
LDPAER    1260#
LDPWRF    1260#
LDSBER    1260#
LECASD    1260#
LEIOPF    1260#
LENXER    1260#
LEPAER    1260#
LEPWRF    1260#
LESBER    1260#
LEUPFW    1260#
LEVNCD    1260#
LEVNPA    1260#
LEVNPD    1260#
LEXCMP    1260#
LFLGCL    1260#
LFLGDS    1260#
LFLGEN    1260#
LFLGST    1260#
LFP       1463#
LGWC       108#
LINSTF    1260#
LINT      1260#
LIOCLR    1260#
LIOPFE    1260#
LKNTRP    1260#
LKTRP     1260#
LLACBL    1260#
LLDUSB    1260#
LLPRCN    1260#
LMUUO     1260#
LMUUOP    1260#
LNXMEN    1260#
LNXMER    1260#
LOGO       101#
LOOPER    1000#
LOOPGM     120#
LOOPTS     121#
LPAREN    1260#
LPARER    1260#
LPDOVT    1260#
LPFWPC    1260#
LPGFTR    1260#
LPICH1    1260#
LPICH2    1260#
```

```
LPICH3    1260#
LPICH4    1260#
LPICH5    1260#
LPICH6    1260#
LPICH7    1260#
LPICHA    1260#
LPICLR    1260#
LPIIP1    1260#
LPIIP2    1260#
LPIIP3    1260#
LPIIP4    1260#
LPIIP5    1260#
LPIIP6    1260#
LPIIP7    1260#
LPIOFF    1260#
LPION     1260#
LPNTRP    1260#
LPRCH1    1260#
LPRCH2    1260#
LPRCH3    1260#
LPRCH4    1260#
LPRCH5    1260#
LPRCH6    1260#
LPRCH7    1260#
LPTRP     1260#
LPWRFE    1260#
LPWRFL    1260#
LREQSE    1260#
LRQCLR    1260#
LSBSEN    1260#
LSBUSE    1260#
LSCASD    1260#
LSECMO    1260#
LSIOPF    1260#
LSMODE    1260#
LSNTRP    1260#
LSNXER    1260#
LSPAER    1260#
LSPWRF    1260#
LSSBER    1260#
LSTRP     1260#
LTRP3T    1260#
LTRPAE    1260#
LTRPEN    1260#
LUSCMP    1260#
LUUO      1260#
LUUO1      454#     1393     1394
LUUO10    1393      1398
LUUO11    1393      1398
LUUO12    1393      1399
LUUO13    1393      1399
LUUO14    1393      1400
LUUO15    1393      1400
```

```
LUUO16    1393      1401
```

```
LUU016     1393     1401
LUU017     1393     1401
LUU02       455#    1393     1395
LUU020     1393     1402
LUU021     1393     1402
LUU022     1393     1403
LUU023     1393     1403
LUU024     1393     1404
LUU025     1393     1404
LUU026     1393     1405
LUU027      458#    1393     1405
LUU03      1393     1395
LUU030     1393     1406
LUU031     1393     1406
LUU032     1393     1407
LUU033     1393     1407
LUU04      1393     1396
LUU05      1393     1396
LUU06       456#    1393     1397
LUU07       457#    1393     1397
LUUOI      1260#
LWRITE     1260#
M1          245#
M10         254#
M11         255#
M12         256#
M13         257#
M14         258#
M15         259#
M16         260#
M17         261#
M18         262#
M19         263#
M2          246#
M3          247#
M4          248#
M5          249#
M6          250#
M7          251#
M8          252#
M9          253#
MAPNEW     1414#
MARGIN     1336#
MBCN         58#
MBERR       418#
MBUS        394#
MCBUS       132#
MCNVER       24#    1308
MEMLOW     1416#
MEMMAP       74#    1314
MEMSIZ     1417#
MEMTOT     1415#
MINUS      1458#
```

```
MLAST       290#
MLAST      290#
MMPROC     131#
MODDVC    1008#
MODDVL    1207     1207#    1309
MODDVU    1208     1208#    1310
MODLNK    1027#    1273
MONCTL    1326#
MONFLG    1325#
MONTEN    1327#
MONTYP    1540#
MPER       102#
MPROC      390#
MPRUN      436#
MPVU      1260#
MSG        465#
MVRERR     430#
NDCB       405#
NDMP       404#
NEXM       105#
NIPORT     129#
NOPNT      996#
NOTWCO     104#
NXMU      1260#
OPRSEL    1010#
OPTIME    1428#
OVN        110#
P           59#    1053     1054     1055     1056     1260#    1348
PAG       1260#
PALERS    1002#
PARCLR    1260#
PARDIS    1260#
PAREA1      90#    1304
PAREA2      91#    1305
PAREA3      92#    1302
PAREA4      93#    1303
PAREA5      94#    1315
PAREA6      95#    1316
PAREA7    1315#
PAREA8    1316#
PARENB    1260#
PARU      1260#
PASCNT    1330#
PAT         56#
PCLEAR     422#
PDISF     1425#
PDLOVU    1260#
PDSABL     434#
PENABL     435#
PERIOD    1455#
PFSTRT    1282#
PGMNAM       8#    1307
PIA00      437#
PIA01      438#
```

```
PIA02      439#
```

```
PIA02       439#
PICHN1     1260#
PICHN2     1260#
PICHN3     1260#
PICHN4     1260#
PICHN5     1260#
PICHN6     1260#
PICHN7     1260#
PICHNA     1260#
PICLR      1260#
PIOFF      1260#
PION       1260#
PLIST      1494#    1494
PLISTE     1494     1496#
PLISTS     1495#
PLUS       1460#
PNTENB     1424#
PNTEXT     1303#
PNTFLG     1423#
PNTINH     1426#
PNTLPT      997#
PNTNAM     1302#
PNTSPC     1427#
PORTP       412#
PPDEC         9#     456
PPDECF        9#     455
PVPAGI     1521#
PWFCLR     1260#
QUEST      1466#
RADIX      1469#
RADLSC     1471#
RADLSP     1470#
RANDBS     1304#
REENTR     1284#
RELIAB     1004#
REPT       1260#
REPT1      1260#
REPTU      1382#
REQSET     1260#
RESQAV      432#
RESRT1     1359#
RESRT2     1360#
RETURN     1291#
RQINT       416#
RSTART      993#
RTP        1464#
RUNALL      122#
RUNFLG     1331#
SADR1        78#    1280
SADR10       87#    1296
SADR11       88#    1297
SADR2        79#    1282
SADR3        80#    1284
```

```
SADR4        81#
```

```
SADR4       81#
SADR5       82#    1345
SADR6       83#    1346
SADR7       84#    1293
SADR8       85#    1294
SADR9       86#    1295
SBINIT    1066     1290#
SCOPE     1383#
SCOPE1       8#     458
SELLAR     425#
SEQ        392#
SFSTRT    1280#
SHWC       109#
SINCYC     426#
SLASH     1467#
SM10      1571#
SPACE     1456#
SPARE1     427#
SPARE2     433#
SRTDDT      81       83      1286#
SSCALL     481#
SSCHK      482#
SSCONT     480#
SSJRST     484#
SSLAST     478#
SSSTRT     479#
START        8#      82        84        85        86        87        88      1274
START1    1293#
START2    1294#
START3    1295#
START4    1296#
START5    1297#
STARTA       8#    1289
SUBLNK    1028#    1290
SWPTAB    1577#
SWTEXR    1305#
SYSEXR    1278#
TAB       1457#
TCALL      493#
TCALLC     494#
TCHECK     496#
TCONT      492#
TDENA      406#
TENLEN     207#
TESTPC    1332#
TEXIT      498#
TICKS     1335#
TJRST      497#
TLAST      490#
TOTALS     994#
TRACET     117#
TSSTAR     499#
TSTART     491#
```

```
TSTEBF     423#
TSTEBF     423#
TTNBRF    1515#
TTYFIL    1431#
TTYSPD    1432#
TUSER      403#
TXALL      470#
TXNOT      468#
TXTINH    1005#
TXYES      469#
USER      1322#
USERF     1260#
USRASB      73#    1312
USRCRF    1441#
USRLFF    1440#
UUODIS    1394#
UUOEXT    1350#
UUORTN    1351#
UUOSKP    1348#
ZALU       399#
ZCBUS      401#
ZEBUS      397#
ZMPROC     400#
ZSEQ       398#
$$MUUO    1546#
$$OUTE    1552#
$$TAX1    1564#
$$TAX2    1565#
$$TOGG    1558#
$$UUO     1535#
$ACCO     1373#
$BEND1    1343#
$BEND2    1344#
$CHRIN    1434#
$CRLF     1436#
$DEVCH    1260#
$DSKUP    1313#
$DVCH1    1260#
$DVOFF    1430#
$EMODE    1311#
$FFF      1438#
$IBUF     1486#
$INEXT    1488#
$INNM     1487#
$ITRHL    1354#
$ITRX1    1355#
$LPAPE      76#
$MMAP     1314#
$MODVL    1309#
$MODVU    1310#
$OBUF     1477#
$ONETM    1337#
$OUTEX    1479#
$OUTNM    1478#
```

```
$PAPER    1260
```

```
$PAPER     1260
$PARER     1361#
$PNAME     1307#
$PSHER     1365#
$PVER      1308#
$RSRTX     1357#
$RSRTY     1358#
$SPAG1     1377#
$START     1274#
$SVAPR     1375#
$SVPAG     1376#
$SVPI      1374#
$SVUPC     1380#
$SVUUO     1379#
$TABF      1437#
$TTCHR     1433#
$TWCNT     1429#
$TYPNB     1435#
$UMODE     1312#
$UORTX     1352#
$USRHL     1356#
$UUOER     1353#      1394
$VTF       1439#
%ACTFL     1386#
%ACTUL     1387#
%COREC     1385#
%CORFL     1384#
%DISCR     1388#
%ERHI1     1527#
%ERHI2     1528#
%ERHI3     1529#
.JB41      1260#
.JBAPR     1260#
.JBCNI     1260#
.JBDDT     1260#
.JBFF      1260#
.JBOPC     1260#
.JBREL     1260#
.JBREN     1260#
.JBSA      1260#
.JBSYM     1260#
.JBTPC     1260#
.JBUSY     1260#
.JBUUO     1260#
.JBVER     1260#
.LA         721#
.LADDR      609#
.LAND       685#
.LB         725#
.LBAD       933#
.LBUS       825#
.LCCAF      761#
.LCCBF      765#
```

```
.LCCCA      741#
```

```
.LCCCA      741#
.LCCCC      753#
.LCCCE      797#
.LCCCP      789#
.LCCEB      757#
.LCCER      773#
.LCCFZ      749#
.LCCGC      745#
.LCCIA      777#
.LCCLW      801#
.LCCMB      781#
.LCCMP      785#
.LCCOF      737#
.LCCON      733#
.LCCPP      793#
.LCCXA      769#
.LCENA      705#
.LCJP       869#
.LCJPP      901#
.LCJS       861#
.LCJV       881#
.LCONT      913#
.LCRTN      897#
.LCRY       849#
.LD         701#
.LDISA      709#
.LENA       713#
.LFUNC      665#
.LJ         613#
.LJMAP      865#
.LJRP       885#
.LJSRP      877#
.LJZ        857#
.LLDCT      905#
.LLDLM      813#
.LLOOP      909#
.LLSAD      805#
.LMARK      929#
.LMGC       625#
.LNAND      689#
.LOENA      621#
.LOP        853#
.LOR        681#
.LPAR       617#
.LPLUS      669#
.LPUSH      873#
.LRAM       717#
.LRDLM      817#
.LRFCT      889#
.LRMIN      677#
.LRPCT      893#
.LSOA       649#
.LSOB       645#
```

```
.LSOQ      641#
.LSAB      637#
.LSAQ      633#
.LSDO      661#
.LSDA      653#
.LSDQ      657#
.LSELC     841#
.LSELE     845#
.LSELF     837#
.LSELM     833#
.LSELP     829#
.LSK       729#
.LSKCN     821#
.LSKMB     809#
.LSMIN     673#
.LSORC     629#
.LSPAR     925#
.LTIM      921#
.LTWB      917#
.LXNOR     697#
.LXOR      693#
.MA        723#
.MADDR     611#
.MAND      687#
.MB        727#
.MBAD      935#
.MBUS      827#
.MCCAF     763#
.MCCBF     767#
.MCCCA     743#
.MCCCC     755#
.MCCCE     799#
.MCCCP     791#
.MCCEB     759#
.MCCER     775#
.MCCFZ     751#
.MCCGC     747#
.MCCIA     779#
.MCCLW     803#
.MCCMB     783#
.MCCMP     787#
.MCCOF     739#
.MCCON     735#
.MCCPP     795#
.MCCXA     771#
.MCENA     707#
.MCJP      871#
.MCJPP     903#
.MCJS      863#
.MCJV      883#
.MCONT     915#
.MCRTN     899#
.MCRY      851#
```

```
.MD         703#
.MDISA      711#
.MENA       715#
.MFUNC      667#
.MJ         615#
.MJMAP      867#
.MJRP       887#
.MJSRP      879#
.MJZ        859#
.MLDCT      907#
.MLDLM      815#
.MLOOP      911#
.MLSAD      807#
.MMARK      931#
.MMGC       627#
.MNAND      691#
.MOENA      623#
.MOP        855#
.MOR        683#
.MPAR       619#
.MPLUS      671#
.MPUSH      875#
.MRAM       719#
.MRDLM      819#
.MRFCT      891#
.MRMIN      679#
.MRPCT      895#
.MSOA       651#
.MSOB       647#
.MSOQ       643#
.MSAB       639#
.MSAQ       635#
.MSDO       663#
.MSDA       655#
.MSDQ       659#
.MSELC      843#
.MSELE      847#
.MSELF      839#
.MSELM      835#
.MSELP      831#
.MSK        731#
.MSKCN      823#
.MSKMB      811#
.MSMIN      675#
.MSORC      631#
.MSPAR      927#
.MTIM       923#
.MTWB       919#
.MXNOR      699#
.MXOR       695#
.PNTOC        9#     454
.RA         722#
.RADDR      610#
```

```
.RAND           686#
.RB             726#
.RBAD           934#
.RBUS           826#
.RCCAF          762#
.RCCBF          766#
.RCCCA          742#
.RCCCC          754#
.RCCCE          798#
.RCCCP          790#
.RCCEB          758#
.RCCER          774#
.RCCFZ          750#
.RCCGC          746#
.RCCIA          778#
.RCCLW          802#
.RCCMB          782#
.RCCMP          786#
.RCCOF          738#
.RCCON          734#
.RCCPP          794#
.RCCXA          770#
.RCENA          706#
.RCJP           870#
.RCJPP          902#
.RCJS           862#
.RCJV           882#
.RCONT          914#
.RCRTN          898#
.RCRY           850#
.RD             702#
.RDISA          710#
.RENA           714#
.RFUNC          666#
.RJ             614#
.RJMAP          866#
.RJRP           886#
.RJSRP          878#
.RJZ            858#
.RLDCT          906#
.RLDLM          814#
.RLOOP          910#
.RLSAD          806#
.RMARK          930#
.RMGC           626#
.RNAND          690#
.ROENA          622#
.ROP            854#
.ROR            682#
.RPAR           618#
.RPLUS          670#
.RPUSH          874#
.RRAM           718#
```

```
.RRDLM      818#
.RRFCT      890#
.RRMIN      678#
.RRPCT      894#
.RSOA       650#
.RSOB       646#
.RSOQ       642#
.RSAB       638#
.RSAQ       634#
.RSDO       662#
.RSDA       654#
.RSDQ       658#
.RSELC      842#
.RSELE      846#
.RSELF      838#
.RSELM      834#
.RSELP      830#
.RSK        730#
.RSKCN      822#
.RSKMB      810#
.RSMIN      674#
.RSORC      630#
.RSPAR      926#
.RTIM       922#
.RTWB       918#
.RXNOR      698#
.RXOR       694#
```

```
ALTCHK     451#
ATABLE     322#
CALC       949#
CFIBF      446#
CFLUSH    1178#
CINVAL    1177#
CLOKOP    1172#
CONCAT     956#
CTABLE     354#
CWRTBI    1179#
DROPDV    1139#
ENDUUO    1185#    1342
EOPUUO    1186#    1344
ERLOOP    1230#
ERLP1     1231#
ERLP2     1232#
ERRHLT    1073#
ERROR     1239#
ERROR1    1250#
ERUUO     1229#
FATAL     1072#
FIELD      962#
FMSG       153#
FMSGC      155#
FMSGCD     161#
FMSGD      158#
FRD36     1214#
FRD8      1215#
FREAD     1213#
FSELEC    1212#
GET       1056#
GO        1053#
HALT        84      85      86      87      88    1058#   1363
ITABLE     371#
JEN       1060#
JRSTF     1059#
JSYS       445#     446
MAPADR    1195#
MAPCNK    1196#
MAPMEM    1192#
MAPPNT    1198#
MAPSET    1197#
MEMSEG    1194#
MEMZRO    1193#
MODPCP    1205#
MODPCU    1204#
MTROP     1173#
MWORD      974#
PBELL     1148#
PCRL      1142#
PCRL2     1146#
PCRL2F    1147#
PCRLF     1143#
```

```
PDEC       450#
PDECF      449#
PFORCE    1150#
PGMINT    1066#
PJRST     1057#
PMSG      1152#
PMSGF     1155#
PNT1      1108#
PNT11     1122#
PNT11F    1123#
PNT1F     1109#
PNT2      1110#
PNT2F     1111#
PNT3      1112#
PNT3F     1113#
PNT4      1114#
PNT4F     1115#
PNT5      1116#
PNT5F     1117#
PNT6      1118#
PNT6F     1119#
PNT7      1120#
PNT7F     1121#
PNTA      1094#
PNTADF    1125#
PNTADR    1124#
PNTAF     1095#
PNTAL     1096#
PNTALF    1097#
PNTCHF    1107#
PNTCHR    1106#
PNTCI     1104#
PNTCIF    1105#
PNTCW     1140#
PNTCWF    1141#
PNTDCF    1133#
PNTDEC    1132#
PNTDS     1134#
PNTDSF    1135#
PNTHW     1128#
PNTHWF    1129#
PNTMGN    1219#
PNTMSF    1101#
PNTMSG    1100#
PNTNM     1136#
PNTOCC     448#
PNTOCF    1131#
PNTOCS    1130#
PNTOCT    1126#
PNTOTF    1127#
PNTSIX    1137#
PNTSXF    1138#
PSIXL     1098#
```

```
PSIXLF    1099#
PSIXLF     1099#
PSIXM      1102#
PSIXMF     1103#
PSP        1144#
PSPF       1145#
PUT        1055#
REPTUO     1233#
RGET        189#
RPUT        175#
RTN        1054#    1350
S           984#
SCOPER      452#
SIXBTZ     1161#
STABLE      306#
SWITCH     1167#
TASCIZ      169#
TEST        209#
TMSG        142#
TMSGC       144#
TMSGCD      150#
TMSGD       147#
TTABLE      338#
TTALTM     1088#
TTICHR     1081#
TTICLR     1091#
TTICNV     1086#
TTIDEC     1085#
TTINO      1083#
TTIOCT     1084#
TTIYES     1082#
TTLOOK     1087#
TTSIXB     1089#
TTYINP     1090#
```

DFPTA1 Test Control and Port Debugger MACRO %53B(1242) 07:46 1-Jul-85 Page 1

```
                                TITLE   DFPTA1  Test Control and Port Debugger

                                SUBTTL  Externs / Entrys

                                SALL
                                .DIREC  FLBLST                  ; First line binary listing only

                                SEARCH  DFPTA,MONSYM
000137                          LOC     137
000137  000000  000004          MCNVER,,DECVER
030600                          LOC     ENDFIX+1

                        ; Miscellaneous stuff

                                INTERN  UDEBUG                  ; no-port user mode debug switch
                                INTERN  MDEBUG                  ; error message debug switch
                                INTERN  SDEBUG                  ; single step debug switch
                                INTERN  LDEBUG                  ; ucode load debug switch
                                INTERN  IDEBUG                  ; fault isolation debug switch
                                INTERN  PGMNAM                  ; program name
                                ENTRY   START,STARTA            ; start addresses
                                INTERN  BUFF                    ; data buffer

                        ; Microcode routines

                                ENTRY   CALPAR,DWCRAM,DRCRAM,TLOAD,MLOADN,MLOADY
                                INTERN  CADDR,CWORDL,CWORDR,PARFLG

                        ; Miscellaneous routines

                                ENTRY   CSRPNT,CSRENG,TRACE,ODELAY,TSTPNT
                                INTERN  TWORD,TPAT,FINPUT,FINECH

                        ; Test variables

                                ENTRY   INDLIS,TSTNAM,TSTCLS,TSTNUM,TSTSUB,TSTPC,TSTADD,TSTFLG
                                ENTRY   TSTMIC,TSTMSK,TSTREP,TSTOFF,SPEAR2,SPREP2
                                INTERN  ALTF,TSLOD1,TSLOD2,TAKFIL,TSTINH,PMODE
```

```

; Test execute routines

        ENTRY   SEXEC,SSPNT             ; single step test
        INTERN  NSSTEP,SSADDR,SEADDR,SAADDR

        ENTRY   AEXEC,AAPNT             ; ALU type test
        INTERN  ALSRT,ALCSR,ALERR,ALCOR,ALACT,ALEBF,ALTIM
        INTERN  ALFLS,ALFLE,ALFLC,ALFLT

        ENTRY   BEXEC,BBPNT             ; basic ALU type test (same as SEXEC
        INTERN  CEBUF,AEBUF             ;  but check EBUF data not next addr)

        ENTRY   TEXEC,TTPNT             ; test (full speed) type
        INTERN  TADDR,TADDRF,TCSRF,TEBUFC,TEBUFA

        ENTRY   CEXEC,CCPNT             ; data transfer type test
        INTERN  CCSR,CLEN,CPAT,CSTATF,CEBUFA,CEBUFC,QADDRF,QADDR,CDERR

        ENTRY   IEXEC,IIPNT             ; interrupt type test
        INTERN  ICSR,IADDR,IFLAG,IIOPF

; Interrupt activity print routines

        ENTRY   IPRINT

; Channel/device handling variables

        ENTRY   CBUF,CSTWRD,PORTCI,PORTNI,NISEL,CISEL,PORSEL,UUT

; EXTERN's located in DFPTA2,3,4,5,6,7,8,9,A.MAC (Tests)

        EXTERN  TSTE1,TSTE2,TSTE3,TSTE4,TSTE5,TSTE6,TSTE7,TSTE10
        EXTERN  TSTE11,TSTE12,TSTE13,TSTE14,TSTE15,TSTE16,TSTE17,TSTE20
        EXTERN  TSTE21,TSTE22,TSTE23,TSTE24,TSTE25,TSTE26,TSTE27,TSTE30
        EXTERN  TSTE31,TSTE32,TSTE33,TSTE34,TSTE35,TSTE36,TSTE37,TSTE40
        EXTERN  TSTE41,TSTE42,TSTE43,TSTE44,TSTE45,TSTE46,TSTE47,TSTE50
        EXTERN  TSTE51,TSTE52,TSTE53,TSTE54,TSTE55,TSTE56,TSTE57,TSTE60
        EXTERN  TSTE61,TSTE62,TSTE63,TSTE64,TSTE65,TSTE66,TSTE67,TSTE70
        EXTERN  TSTE71,TSTE72,TSTE73,TSTE74,TSTE75,TSTE76

        EXTERN  TSTS1,TSTS2,TSTS3,TSTS4,TSTS5,TSTS6,TSTS7,TSTS10
        EXTERN  TSTS11,TSTS12,TSTS13,TSTS14,TSTS15,TSTS16,TSTS17,TSTS20
        EXTERN  TSTS21,TSTS22,TSTS23,TSTS24,TSTS25,TSTS26,TSTS27,TSTS30
        EXTERN  TSTS31,TSTS32,TSTS33,TSTS34,TSTS35,TSTS36,TSTS37,TSTS40
        EXTERN  TSTS41,TSTS42,TSTS43,TSTS44,TSTS45,TSTS46,TSTS47,TSTS50
        EXTERN  TSTS51,TSTS52,TSTS53,TSTS54,TSTS55,TSTS56,TSTS57,TSTS60
        EXTERN  TSTS61,TSTS62,TSTS63,TSTS64,TSTS65,TSTS66,TSTS67,TSTS70

```

```
                EXTERN  TSTA1,TSTA2,TSTA3,TSTA4,TSTA5,TSTA6,TSTA7,TSTA10
                EXTERN  TSTA11,TSTA12,TSTA13,TSTA14,TSTA15,TSTA16,TSTA17,TSTA20
                EXTERN  TSTA21,TSTA22,TSTA23,TSTA24,TSTA25,TSTA26,TSTA27,TSTA30
                EXTERN  TSTA31,TSTA32,TSTA33,TSTA34,TSTA35,TSTA36,TSTA37,TSTA40
                EXTERN  TSTA41,TSTA42

                EXTERN  TSTU1,TSTU2,TSTU3,TSTU4,TSTU5,TSTU6,TSTU7,TSTU10
                EXTERN  TSTU11,TSTU12,TSTU13,TSTU14,TSTU15,TSTU16,TSTU17,TSTU20
                EXTERN  TSTU21,TSTU22,TSTU23,TSTU24,TSTU25,TSTU26,TSTU27,TSTU30
                EXTERN  TSTU31,TSTU32,TSTU33,TSTU34,TSTU35,TSTU36,TSTU37,TSTU40
                EXTERN  TSTU41

                EXTERN  TSTC1,TSTC2,TSTC3,TSTC4,TSTC5,TSTC6,TSTC7,TSTC10
                EXTERN  TSTC11,TSTC12,TSTC13,TSTC14,TSTC15,TSTC16,TSTC17,TSTC20
                EXTERN  TSTC21,TSTC22,TSTC23,TSTC24,TSTC25,TSTC26,TSTC27,TSTC30
                EXTERN  TSTC31,TSTC32,TSTC33,TSTC34,TSTC35,TSTC36,TSTC37,TSTC40
                EXTERN  TSTC41,TSTC42,TSTC43,TSTC44,TSTC45,TSTC46,TSTC47,TSTC50
                EXTERN  TSTC51,TSTC52,TSTC53,TSTC54,TSTC55,TSTC56,TSTC57,TSTC60

; EXTERN's located in DFPTAI.MAC (Port Handling Routines module)

                EXTERN  LDEBUF,LDRAR,LDCRAM,LDCSR,RDEBUF,RDLAR,RDCRAM,RDCSR
                EXTERN  SETEBU,SETLAR,CLREBU,CHKCSR,ERESET,IPACLR,ISTOP
                EXTERN  IPASRT,IPASTP,IPASST,SLAST,SNEXT,EADDR,AADDR,SDATA
                EXTERN  INITPI,INITPD,.PIOFF,.PION,SETVEC,.CONI,.CONO,.DATAO,.DATAI
                EXTERN  INTNUM,INTPC,INTTYP,INTCON,INTAPR,INTTIM,INTUSE,INTCSR,INTEND
                EXTERN  CHINIT,GENCCW,GETLOG,CCWPNT,LOGPNT,CBASE,LOGBUF,CHDATA
                EXTERN  DSETUP,DEVREQ,DEVREL,USRION,PFSET,.INPNT,.INWD1
                EXTERN  HAVENI,HAVECI,REQ1,REQ2,REL

; EXTERN's located in DFPTAU.MAC (Utility Routines module)

                EXTERN  CLRBUF,TTYYES,STCLOK,RUNTME,CURTIM,DEBTIM,PNTDEV,.CLOSE
                EXTERN  FORPNT,CONVSX,FINIT,UNAME,FGETW,FINCMD,FIOFF,FOARG,.RESET
                EXTERN  .SWCHP,SWITT,.ISWT,SWCHPT,.RSWIT,.LSWIT,SWRGT,SWCOM,.PNTOC
                EXTERN  .COMM,.CGOT,.SARG,.OARG,.DARG,ARGFLG,ARGUM,CHKARG,LASARG,DECYN
                EXTERN  SCOOFF,SCOSW,BUFGEN,BUFCOM,WRDERR,PATPNT

```

```
                                                 SUBTTL  Initialization

                                                 SALL

030600                                   PGMNAM: ASCIZ   $
030600  015 012 104 106 120              DFPTA Port Basic Diagnostic
                                         $

030607  402 00 0 00 030056               SB:     SETZM   $ONETM                  ; reinit everything
030610                                   START:

                                         ; First clear debug switches and set up PGMINT variables - MAPNEW, PVPAGI

030610  402 00 0 00 064654'                      SETZM   UDEBUG#                 ; no-port user mode debug switch
030611  402 00 0 00 064614'                      SETZM   MDEBUG#                 ; error message debug switch
030612  402 00 0 00 064641'                      SETZM   SDEBUG#                 ; single step debug switch
030613  402 00 0 00 064610'                      SETZM   LDEBUG#                 ; ucode load debug switch
030614  402 00 0 00 064606'                      SETZM   IDEBUG#                 ; fault isolation debug switch
030615  476 00 0 00 030143                       SETOM   MAPNEW                  ; map all memory
030616  476 00 0 00 030511                       SETOM   PVPAGI                  ; always use paging

                                         ; MCPUS - Multiple cpu print switch - If this flag is cleared, the multiple
                                         ; cpu message is printed in full, if set, only the question asking which
                                         ; cpu to run on is asked.

030617  336 00 0 00 030056                       SKIPN   $ONETM                  ; first time through here?
030620  402 00 0 00 064613'                      SETZM   MCPUS#                  ; yes - clear the flag

                                         ; Set up cache usage

030621  402 00 0 00 030506                       SETZM   CSHFLG                  ; allow pgm and any buffer space
030622  402 00 0 00 030507                       SETZM   CSHMEM                  ;   to be in cache

                                         ; Init file input switch

030623  402 00 0 00 064603'                      SETZM   FINPUT#                 ; initialize 'file input' switch
030624  476 00 0 00 064602'                      SETOM   FINECH#                 ; set echo switch
```

```

                                        ; Init flags/switches

030625  402 00 0 00 064605'                     SETZM   FMTTYP#                 ; allow all data formats
030626  402 00 0 00 064571'                     SETZM   BONETM#                 ; clear $ONETM flag
030627  336 00 0 00 030056                      SKIPN   $ONETM                  ; 1st time pgm init?
030630  476 00 0 00 064571'                     SETOM   BONETM                  ; yes - set $ONETM flag
030631  332 00 0 00 030056                      SKIPE   $ONETM                  ; 1st time pgm init?
030632  254 00 0 00 030644                      JRST    STRT1                   ; no - skip next bit of code

                                        ; First time only code

030633  402 00 0 00 000000*                     SETZM   SWRGT                   ; yes - clear right hand switches
030634  201 00 0 00 000000*                     MOVEI   .RSWIT                  ; get start addr of prompt table
030635  202 00 0 00 030525                      MOVEM   SWPTAB                  ; setup for switch prompting

                                        ; Init print enable & forced print switch, and SPEAR flags

030636  211 00 0 00 023420                      MOVNI   ^D10000                 ; set the printout limit
030637  202 00 0 00 030220                      MOVEM   PNTENB                  ; so won't die too soon
030640  402 00 0 00 000000*                     SETZM   FORPNT                  ; clear forced print switch
030641  402 00 0 00 030521                      SETZM   $$TOGGLE                ; enable switches
030642  402 00 0 00 064643'                     SETZM   SPEAR1#                 ; clear 'SPEAR initial msg prnted' flag
030643  402 00 0 00 064644'                     SETZM   SPEAR2#                 ; clear 'SPEAR err/fin msg prnted' flag

                                        ; Standard startup and handle paging and clock

030644  260 17 0 00 000000*             STRT1:  GO      .RESET                  ; reset everything
030645  265 00 0 00 030011                      PGMINT                          ; init the program
030646  205 00 0 00 500000                      MOVSI   500000                  ; access bits for the page table
030647  502 00 0 00 000400                      HLLM    400                     ; maps 340000 to physical 0
030650  201 00 0 00 340000                      MOVEI   340000                  ; causes paging when E is calculated
030651  202 00 0 00 000000*                     MOVEM   CBASE                   ; channel routines now page correctly
030652  260 17 0 00 000000*                     GO      STCLOK                  ; start the clock...
030653  402 00 0 00 030047                      SETZM   PASCNT                  ; clear pass count

                                        ; Initialize page fault handler

030654  260 17 0 00 000000*                     GO      PFSET                   ; set up for page faults on EBUS xfrs

                                        ; Set/clear debug switches

030655  402 00 0 00 064654'                     SETZM   UDEBUG                  ; clear no-port user mode debug switch
030656  402 00 0 00 064614'                     SETZM   MDEBUG                  ; clear error message debug switch
030657  402 00 0 00 064610'                     SETZM   LDEBUG                  ; clear ucode load debug switch
030660  402 00 0 00 064606'                     SETZM   IDEBUG                  ; fault isolation debug switch
```

```
030661  260 17 0 00 000000*          GO      SWITT                     ; get switches
030662  602 00 0 00 002000           TRNE    DEBUG1                    ; debug switch set?
030663  476 00 0 00 064654'          SETOM   UDEBUG                    ; set no-port user mode debug switch
030664  602 00 0 00 001000           TRNE    DEBUG2                    ; debug switch set?
030665  476 00 0 00 064614'          SETOM   MDEBUG                    ; set error message debug switch
030666  602 00 0 00 000200           TRNE    DEBUG4                    ; debug switch set?
030667  476 00 0 00 064610'          SETOM   LDEBUG                    ; set ucode load debug switch
030670  602 00 0 00 000100           TRNE    DEBUG5                    ; debug switch set?
030671  476 00 0 00 064606'          SETOM   IDEBUG                    ; set fault isolation debug switch
030672  201 00 0 00 000005           MOVEI   ITERAT                    ; get default pass count
030673  202 00 0 00 030024           MOVEM   ITRCNT                    ; save in FIXED area location
030674  402 00 0 00 047034           SETZM   PMODE                     ; set program mode to 'no test'

                             ; Determine configuration

030675  260 17 0 00 032314           GO      CONFIG                    ; determine configuration
030676  260 17 0 00 030661*          GO      SWITT                     ; get program switches
030677  607 00 0 00 000010           TLNN    OPRSEL                    ; operator select switch set?
                                     JRST    [GO      TSTSEX           ; no - print selection and
                                              MOVE    NISEL            ;   continue.  If no ports
                                              IOR     CISEL            ;   are selected, exit.
                                              TRNN    4
                                              JRST    STARTB
030700  254 00 0 00 052515                    JRST    STARTA]
030701  602 00 0 00 000040           TRNE    NIPORT                    ; NI port selected explicitly?
                                     GO      [MOVE    1,NISEL          ; yes - get NI select word
                                              TRNN    1,4              ; actually selected?
                                              TRZ     NIPORT           ; no - clear switch
030702  260 17 0 00 052523                    RTN]
030703  602 00 0 00 000020           TRNE    CIPORT                    ; CI port selected explicitly?
                                     GO      [MOVE    1,CISEL          ; yes - get CI select word
                                              TRNN    1,4              ; actually selected?
                                              TRZ     CIPORT           ; no - clear switch
030704  260 17 0 00 052527                    RTN]
030705  606 00 0 00 000060           TRNN    NIPORT!CIPORT             ; any ports selected?
030706  260 17 0 00 032457           GO      TSTSEG                    ; no - ask what devices to test

                             ; Handle SPEAR reporting

030707  260 17 0 00 044171           GO      SPREP1                    ; do initial SPEAR report
```

```



                                         ; Go do a test

030710  260 17 0 00 000000*     STARTA: GO      INITPI                  ; initialize PI system
030711  260 17 0 00 000000*             GO      SETVEC                  ; set up vector addresses
030712  260 17 0 00 030731              GO      DISPAT                  ; do a test
030713  332 00 0 00 064570'             SKIPE   ALTF                    ; altmode typed?
030714  254 00 0 00 030727              JRST    STARTB                  ; yes - exit
030715  260 17 0 00 030676*             GO      SWITT                   ; get switches
030716  336 00 0 00 030037              SKIPN   USER                    ; user mode?
030717  607 00 0 00 400000              TLNN    ABORT                   ; abort switch set?
030720  334 00 0 00 000000              SKIPA                           ; no (or user mode) - continue
030721  254 00 0 00 030727              JRST    STARTB                  ; yes - exit
030722  037 12 0 00 000004              ENDUUO                          ; end of pass handling
030723  254 00 0 00 030710              JRST    STARTA                  ; ITRCNT not exhausted - continue
030724  260 17 0 00 030715*             GO      SWITT                   ; get switches
030725  602 00 0 00 040000              TRNE    LOOPGM                  ; loop on program switch set?
030726  254 00 0 00 030710              JRST    STARTA                  ; yes - loop forever
030727  260 17 0 00 000000*     STARTB: GO      DEVREL                  ; Release whichever device we had
030730  037 16 0 00 000004              EOPUUO                          ; ITRCNT exhausted - exit pgm
```

```
        SUBTTL  Test/Command Selection

;#*******************************************************************
;* DISPAT - Routine to dispatch to proper test as typed by the operator
;*          in response to the 'WHAT TEST' prompt.
;
; COMMAND       DEFINITION
; -------       ----------
; ALL    n      All tests
; EBUS   n      EBus module tests
; SEQ    n      SEQ related tests
; ALU    n      2901 related tests
; MPROC  n      MPROC module tests
; CBUS   n      CBus module tests
; MBUS   n      MBUS related tests
; XXn    n      Test n, category XX (where XX can be EB,SE,AL,MP,CB)
;
; Other tests:
;
; DEBUG         Enter port debugger
;
; CONFIG        Determine/print configuration
; CONPNT        Print configuration
; SELECT        Select devices
; DIAMON        Enter DIAMON
; DDT           Enter DDT
; HELP          Print this text
; EXIT          Exit program
;
; TAKE          Take commands from a file
; NTAKE         Take commands from a file (without echoing them)
;
; ENABLE        Enable execution of a test
; DISABL        Disable execution of a test
; DISLIS        List disabled tests
;
; LIST n        List all tests
; LISTEB n      List EBus module tests
; LISTMP n      List MPROC module tests
; LISTCB n      List CBus module tests
; LISTSE n      List Sequencer related tests
; LISTAL n      List 2901's related tests
;
; Switches:  To complement switch, type switch name
;
; SWITCH        Enter switches
; SWPRIN        Print current switches
;
; NOPNT   LOOPER  RELIAB   TRACE   LOOPGM  DSPEAR  SDEBUG  NIPORT  MMPROC
; PNTLPT  ERSTOP  TXTINH   INHFLT  LOOPTS  UDEBUG  LDEBUG  CIPORT  MCBUS
; DING    PALERS  OPRSEL   INHMSG  RUNALL  MDEBUG  IDEBUG
;#*******************************************************************
```

```

030731  402 00 0 00 064570'     DISPAT: SETZM   ALTF#                   ; clear 'altmode typed' flag

030732  260 17 0 00 030724*             GO      SWITT                   ; get switches
030733  603 00 0 00 000010              TLNE    OPRSEL                  ; OPRSEL switch set?
030734  037 01 0 00 052533              FMSGCD  <Type ? for help>       ; yes - print this

                                ; Do a little initialization - first set up altmode transfer address

030735  260 17 0 00 030732*     DISPO:  GO      SWITT                   ; get switches
030736  603 00 0 00 000010              TLNE    OPRSEL                  ; OPRSEL switch set?
                                        MOVEI   1,[PCRLF                ; yes - set $ addr to print blank line
030737  201 01 0 00 052537                         JRST DISPO]          ;  line and continue at 'What test'
030740  607 00 0 00 000010              TLNN    OPRSEL                  ; OPRSEL switch set?
                                        MOVEI   1,[SETZM ALTF           ; no - set up $ xfr addr to exit pass
                                                   GO     FIOFF
030741  201 01 0 00 052541                         RTN]
030742  202 01 0 00 030063              MOVEM   1,ALTMGO                ; store it
030743  402 00 0 00 064570'             SETZM   ALTF                    ; clear 'altmode typed' flag

                                ; Set up print limit, clear error flag, clear input buffer, set up MBCN

030744  201 00 0 00 011610              MOVEI   ^D5000                  ; re-set up print limit just in case
030745  212 00 0 00 030220              MOVNM   PNTENB                  ;   the last test exceeded it
030746  400 15 0 00 000000              SETZ    ERFLG,                  ; clear error flag
030747  260 17 0 00 000000*             GO      CLRBUF                  ; clear input buffer
030750  476 00 0 00 064646'             SETOM   TSLOD1#                 ; initialize ucode loaded flag (NI)
030751  476 00 0 00 064647'             SETOM   TSLOD2#                 ; initialize ucode loaded flag (CI)

                                ; Set/clear debug switches

030752  402 00 0 00 064654'             SETZM   UDEBUG                  ; clear no-port user mode debug switch
030753  402 00 0 00 064614'             SETZM   MDEBUG                  ; clear error message debug switch
030754  402 00 0 00 064641'             SETZM   SDEBUG                  ; clear single step debug switch
030755  402 00 0 00 064610'             SETZM   LDEBUG                  ; clear ucode load debug switch
030756  402 00 0 00 064606'             SETZM   IDEBUG                  ; fault isolation debug switch
030757  260 17 0 00 030735*             GO      SWITT                   ; get switches
030760  602 00 0 00 002000              TRNE    DEBUG1                  ; debug switch set?
030761  476 00 0 00 064654'             SETOM   UDEBUG                  ; set no-port user mode debug switch
030762  602 00 0 00 001000              TRNE    DEBUG2                  ; debug switch set?
030763  476 00 0 00 064614'             SETOM   MDEBUG                  ; set error message debug switch
030764  602 00 0 00 000400              TRNE    DEBUG3                  ; debug switch set?
030765  476 00 0 00 064641'             SETOM   SDEBUG                  ; set single step debug switch
030766  602 00 0 00 000200              TRNE    DEBUG4                  ; debug switch set?
030767  476 00 0 00 064610'             SETOM   LDEBUG                  ; set ucode load debug switch
030770  602 00 0 00 000100              TRNE    DEBUG5                  ; debug switch set?
030771  476 00 0 00 064606'             SETOM   IDEBUG                  ; set fault isolation debug switch
030772  402 00 0 00 047034              SETZM   PMODE                   ; set program mode to 'no test'
030773  402 00 0 00 064616'             SETZM   MULFLG#                 ; clear 'multiple examine' flag
030774  476 00 0 00 064617'             SETOM   MULINI#                 ; set 'initial examine' flag
```

```

                                    ; Set up PI system in case any interrupts occur

030775  260 17 0 00 030710*                   GO      INITPI                   ; init PI system
030776  260 17 0 00 030711*                   GO      SETVEC                   ; set up vector addresses

                                    ; Do ALL test automatically if OPRSEL switch is not set

030777  201 01 0 00 000001                    MOVEI   1,1                      ; set to ALL test
031000  607 00 0 00 000010                    TLNN    OPRSEL                   ; OPRSEL switch set?
031001  254 00 0 00 031067                    JRST    DISP2                    ; no - go do test
031002  402 00 0 00 030047                    SETZM   PASCNT                   ; clear pass count

                                    ; Now handle a normal command

031003  402 00 0 00 000000*                   SETZM   .INWD1                   ; clear special interrupt data
031004  476 00 0 00 064571'                   SETOM   BONETM                   ; set 'not first time' flag
031005  336 00 0 00 064603'                   SKIPN   FINPUT                   ; file input?
031006  476 00 0 00 064602'                   SETOM   FINECH                   ; no - ensure echo flag set
031007  332 00 0 00 064602'                   SKIPE   FINECH                   ; echo flag set?
                                              GO      [PFORCE                  ; no - print all this
                                                       PCRLF
                                                       FMSG    <What test - >
031010  260 17 0 00 052547                             RTN]
031011  332 00 0 00 064603'                   SKIPE   FINPUT                   ; file input?
                                              JRST    [GO      FINCMD          ; yes - input sixbit cmd
                                                       JRST    DISP0           ; error or EOF
031012  254 00 0 00 052553                             JRST    DISP0B]         ; continue
031013  037 10 0 00 000003          DISP0A:   TTSIXB                           ; get the name in sixbit
                                              ALTCHK  [GET     XXW#            ; adjust stack properly
                                                       SKIPN   $TWCNT          ; did a timeout occur?
                                                       JRST    DISP0           ; yes - keep looking for input
                                                       MOVE    1,$TTCHR        ; no - get character typed
                                                       CAIN    1,40            ; space (delimiter) typed?
                                                       JRST    .+1             ; yes - continue (no error condition)
                                                       CAIN    1,77            ; was a question mark typed?
                                                       JRST    DISQUE          ; yes - handle
031014  007 00 0 00 052556                             JRST    DISPE]          ; no - error - reask question
                                    DISP0B:   JUMPE   [MOVE    $TTCHR          ; anything typed?  if a space, keep
                                                       CAIN    40              ;   looking for an argument, if not
                                                       JRST    DISP0A          ;   exit
031015  322 00 0 00 052567                             JRST    DISP0]
031016  202 00 0 00 064653'                   MOVEM   TSTSAV#                  ; save the typed name
031017  402 00 0 00 000000*                   SETZM   ARGFLG                   ; clear 'argument given' flag
031020  200 01 0 00 030231                    MOVE    1,$TTCHR                 ; get last character typed
031021  302 01 0 00 000015                    CAIE    1,15                     ; CR? (end of line)
031022  476 00 0 00 031017*                   SETOM   ARGFLG                   ; set 'argument given' flag
```

```

                                    ; Process command

031023  200 01 0 00 064653'         DISP1:  MOVE    1,TSTSAV                ; get command into AC1
031024  201 02 0 00 031304                  MOVEI   2,DISLIS                ; get address of cmd list
031025  260 17 0 00 000000*                  GO      .COMM                   ; handle command decoding
031026  334 00 0 00 000000                    SKIPA                          ; error return
031027  254 00 0 00 031067                   JRST    DISP2                   ; dispatch to test

                                    ; Now check to see if a test number was selected in the form
                                    ; XXnnn where XX is the test class and nnn is the test number.

031030  260 17 0 00 034315                   GO      TSTDEC                  ; check for alternate format
031031  254 00 0 00 031051                    JRST   DISPD                   ; error return (unrecognizable)
031032  254 00 0 00 031036                    JRST   DISPR                   ; error return (range error)
031033  370 00 0 00 047015                   SOS     TSTADD                  ; good - adjust for SELTST
031034  260 17 0 00 032751                   GO      TSTSPC                  ; execute it
031035  254 00 0 00 031070                   JRST    DISPX                   ; end of test

                                    ; Range error

031036  202 01 0 00 064637'         DISPR:  MOVEM   1,SAVMAX#               ; save maximum test number
031037  037 01 0 00 052573                   FMSGC   <? >                    ; print error message
031040  200 01 0 00 047010                   MOVE    1,TSTCLS
031041  200 00 0 01 031305                   MOVE    DISLIS+1(1)
031042  037 01 0 00 000002                   PNTSXF
031043  037 01 0 00 052574                   FMSG    < Test selected is not in range 1->
031044  200 00 0 00 064637'                  MOVE    SAVMAX                  ; get range
031045  037 17 0 00 000003                   PNTOCF                          ; print it
031046  037 01 0 00 030242                   PCRLF
031047  260 17 0 00 000000*                  GO      FIOFF                   ; turn off file input
031050  254 00 0 00 030735                   JRST    DISPO                   ; continue

                                    ; Possibly a DEBUG command was typed - check for this and handle

031051  200 01 0 00 064653'         DISPD:  MOVE    1,TSTSAV                ; get command into AC1
031052  201 02 0 00 034776                   MOVEI   2,.DBCMD                ; get address of cmd list
031053  260 17 0 00 031025*                  GO      .COMM                   ; handle command decoding
031054  254 00 0 00 031064                    JRST   DISPE                   ; unrecognizable
031055  260 17 0 00 032520                   GO      PTEST                   ; determine port to use
031056  260 17 0 00 032537                   GO      PTESTP                  ; print results
031057  322 16 0 00 030735                   JUMPE   MBCN,DISPO              ; exit if no ports are selected
031060  260 17 0 00 000000*                  GO      DEVREQ                  ; ensure the device is selected
031061  334 00 0 00 000000                   SKIPA                           ; error - continue
031062  260 17 1 01 035146                   GO      @.DBDIS(1)              ; dispatch to test
031063  254 00 0 00 030735                   JRST    DISPO                   ; return to 'DEBUG>' prompt

                                    ; Unrecognized command

031064  037 01 0 00 052603          DISPE:  FMSGCD  <? Unrecognizable - Type ? for help>
031065  260 17 0 00 031047*                  GO      FIOFF                   ; turn off file input
031066  254 00 0 00 030735                   JRST    DISPO
```

```
                                       ; Dispatch to test

031067  260 17 1 01 031374             DISP2:  GO      @DISGO(1)               ; dispatch to test

                                       ; End of command - if OPRSEL switch set, ask for command
                                       ;                  if OPRSEL switch not set, it is end of pass

031070  260 17 0 00 030757*            DISPX:  GO      SWITT                   ; get program switches
031071  603 00 0 00 000010                     TLNE    OPRSEL                  ; OPRSEL set?
031072  254 00 0 00 030735                     JRST    DISP0                   ; yes - not really end of pass
031073  263 17 0 00 000000                     RTN                             ; no - end of pass

                                       ; Handle ? typed

031074  037 01 0 00 000026             DISQUE: PFORCE                          ; handle Control-O
031075  037 00 0 00 031077                     PNTMSG  DISQUU                  ; print text
031076  254 00 0 00 030735                     JRST    DISP0                   ; continue

031077                                 DISQUU: ASCIZ   /
031077  015 012 124 145 163            Tests       Other           Switches
                                       -----   -------------   ----------------------
                                       ALL     ENABLE  LIST    NOPNT   TRACE   UDEBUG
                                       EBUS    DISABL  LISTEB  PNTLPT  INHFLT  MDEBUG
                                       SEQ     DISLIS  LISTSE  DING    INHMSG  SDEBUG
                                       ALU             LISTAL  LOOPER  LOOPGM  LDEBUG
                                       MPROC   CONFIG  LISTMP  ERSTOP  LOOPTS  IDEBUG
                                       CBUS    CONPNT  LISTCB  PALERS  RUNALL  NIPORT
                                       MBUS    SELECT  TAKE    RELIAB  TXTINH  CIPORT
                                                       NTAKE   OPRSEL          DSPEAR
                                       DEBUG   DIAMON  HELP            SWITCH  MMPROC
                                               DDT     EXIT            SWPRIN  MCBUS

                                       (Or XXn where n is test #, and XX is test category EB,SE,AL,MP,CB)
                                       /



                                       ;#*****************************************************************
                                       ;* INDLIS - List of classes that may be specified with individual test numbers
                                       ;#*****************************************************************

031275  45 42 00 00 00 00              INDLIS: SIXBIT  /EB/                    ; an EBus module tests
031276  63 45 00 00 00 00                      SIXBIT  /SE/                    ; an Sequencer related tests
031277  41 54 00 00 00 00                      SIXBIT  /AL/                    ; an 2901's related tests
031300  55 60 00 00 00 00                      SIXBIT  /MP/                    ; an MPROC module tests
031301  43 42 00 00 00 00                      SIXBIT  /CB/                    ; an CBus module tests
031302  55 42 00 00 00 00                      SIXBIT  /MB/                    ; an MBUS related tests
031303  000 00 0 00 000000                     Z                               ; list terminator
```

```
                                  ;#*******************************************************************
                                  ;* DISLIS -- List of all valid test names that can be typed
                                  ;#*******************************************************************

031304  777777  777655            DISLIS: -123                            ; ignore
031305  41 54 54 00 00 00                 SIXBIT  /ALL/                   ; execute all tests
031306  45 42 65 63 00 00                 SIXBIT  /EBUS/                  ; execute EBus module tests
031307  63 45 61 00 00 00                 SIXBIT  /SEQ/                   ; execute Sequencer related tests
031310  41 54 65 00 00 00                 SIXBIT  /ALU/                   ; execute 2901's related tests
031311  55 60 62 57 43 00                 SIXBIT  /MPROC/                 ; execute MPROC module tests
031312  43 42 65 63 00 00                 SIXBIT  /CBUS/                  ; execute CBus module tests
031313  55 42 65 63 00 00                 SIXBIT  /MBUS/                  ; execute MBUS related tests
031314  62 41 56 44 57 55                 SIXBIT  /RANDOM/                ; execute tests randomly

                                  ; Other tests/switches ...

031315  64 41 53 45 00 00                 SIXBIT  /TAKE/                  ; take commands from a file
031316  56 64 41 53 45 00                 SIXBIT  /NTAKE/                 ; take commands from a file
031317  44 45 42 65 47 00                 SIXBIT  /DEBUG/                 ; execute debugger test
031320  43 57 56 46 51 47                 SIXBIT  /CONFIG/                ; determine configuration
031321  43 57 56 60 56 64                 SIXBIT  /CONPNT/                ; print configuration
031322  63 45 54 45 43 64                 SIXBIT  /SELECT/                ; select device(s) to test
031323  44 51 41 55 57 56                 SIXBIT  /DIAMON/                ; exit to DIAMON
031324  44 44 64 00 00 00                 SIXBIT  /DDT/                   ; enter DDT
031325  45 70 51 64 00 00                 SIXBIT  /EXIT/                  ; exit pgm
031326  50 45 54 60 00 00                 SIXBIT  /HELP/                  ; help

031327  45 56 41 42 54 45                 SIXBIT  /ENABLE/                ; enable execution of a test
031330  44 51 63 41 42 54                 SIXBIT  /DISABL/                ; disable execution of a test
031331  44 51 63 54 51 63                 SIXBIT  /DISLIS/                ; list disabled tests

031332  54 51 63 64 00 00                 SIXBIT  /LIST/                  ; list all tests
031333  54 51 63 64 45 42                 SIXBIT  /LISTEB/                ; list EBus module tests
031334  54 51 63 64 63 45                 SIXBIT  /LISTSE/                ; list Sequencer related tests
031335  54 51 63 64 41 54                 SIXBIT  /LISTAL/                ; list 2901's related tests
031336  54 51 63 64 55 60                 SIXBIT  /LISTMP/                ; list MPROC module tests
031337  54 51 63 64 43 42                 SIXBIT  /LISTCB/                ; list CBus module tests

031340  63 67 51 64 43 50                 SIXBIT  /SWITCH/                ; input switches
031341  63 67 60 62 51 56                 SIXBIT  /SWPRIN/                ; print switch selections

031342  56 57 60 56 64 00                 SIXBIT  /NOPNT/                 ; don't print anything
031343  60 56 64 54 60 64                 SIXBIT  /PNTLPT/                ; print on printer
031344  44 51 56 47 00 00                 SIXBIT  /DING/                  ; ding on error
031345  54 57 57 60 45 62                 SIXBIT  /LOOPER/                ; loop on error
031346  45 62 63 64 57 60                 SIXBIT  /ERSTOP/                ; halt on error
031347  60 41 54 45 62 63                 SIXBIT  /PALERS/                ; print all errors
031350  62 45 54 51 41 42                 SIXBIT  /RELIAB/                ; reliability testing
031351  64 70 64 51 56 50                 SIXBIT  /TXTINH/                ; shorten printing
031352  57 60 62 63 45 54                 SIXBIT  /OPRSEL/                ; operator select
```

```
031353  64 62 41 43 45 00          SIXBIT  /TRACE/                    ; trace pgm flow
031354  51 56 50 46 54 64          SIXBIT  /INHFLT/                   ; inhibit fault isolation
031355  51 56 50 55 63 47          SIXBIT  /INHMSG/                   ; inhibit error messages
031356  54 57 57 60 47 55          SIXBIT  /LOOPGM/                   ; loop on program
031357  54 57 57 60 64 63          SIXBIT  /LOOPTS/                   ; loop on test
031360  62 65 56 41 54 54          SIXBIT  /RUNALL/                   ; run all test segments
031361  44 63 60 45 41 62          SIXBIT  /DSPEAR/                   ; disable SPEAR reporting
031362  65 44 45 42 65 47          SIXBIT  /UDEBUG/                   ; no-port user mode debug switch
031363  55 44 45 42 65 47          SIXBIT  /MDEBUG/                   ; error message debug switch
031364  63 44 45 42 65 47          SIXBIT  /SDEBUG/                   ; single step debug switch
031365  54 44 45 42 65 47          SIXBIT  /LDEBUG/                   ; ucode load debug switch
031366  51 44 45 42 65 47          SIXBIT  /IDEBUG/                   ; fault isolation debug switch
031367  56 51 60 57 62 64          SIXBIT  /NIPORT/                   ; NI port selected
031370  43 51 60 57 62 64          SIXBIT  /CIPORT/                   ; CI port selected
031371  55 55 60 62 57 43          SIXBIT  /MMPROC/                   ; missing Mproc module
031372  55 43 42 65 63 00          SIXBIT  /MCBUS/                    ; missing Cbus module
031373  000 00 0 00 000000         Z



                           ;#*********************************************************************
                           ;* DISGO - Test dispatch addresses
                           ;#*********************************************************************

031374  000000  052526     DISGO:  [RTN]                              ; ignore
031375  000000  031506             TSTEXE                             ; ALL - execute all tests
031376  000000  031506             TSTEXE                             ; EBUS - execute EBus module tests
031377  000000  031506             TSTEXE                             ; SEQ - execute SEQ related tests
031400  000000  031506             TSTEXE                             ; ALU - execute 2901 related tests
031401  000000  031506             TSTEXE                             ; MPROC - execute MPROC module tests
031402  000000  031506             TSTEXE                             ; CBUS - execute CBus module tests
031403  000000  031463             TSTMBU                             ; MBUS - execute MBUS related tests
031404  000000  033015             TSTRAN                             ; RANDOM - execute tests randomly

031405  000000  032715             TSTTAK                             ; TAKE - take commands from a file
031406  000000  032713             TSTTAN                             ; NTAKE - take commands from a file
031407  000000  034365             TSTDEB                             ; DEBUG - execute debugger test
                                   [GO      CONFIG                    ; CONFIG - determine configuration
                                    GO      TSTSEL
031410  000000  052613              RTN]
                                   [MOVE    1,[FMSGCD (CONPNT <CR>)]; CONPNT - print configuration
                                    GO      CHKARG
                                    RTN
                                    GO      CONPNT
031411  000000  052623              RTN]
031412  000000  032416             TSTSEL                             ; SELECT - select port to test
031413  000000  032265             TSTDIA                             ; DIAMON - re-enter DIAMON
031414  000000  032272             TSTDDT                             ; DDT - enter DDT
031415  000000  032303             TSTHLT                             ; EXIT - exit
031416  000000  032554             TSTHLP                             ; HELP - help
031417  000000  032616             TSTENA                             ; ENABLE - enable test
031420  000000  032612             TSTDSA                             ; DISABL - disable test
031421  000000  032644             TSTDSL                             ; DISLIS - list disabled tests
```

```
    621
    622                        000026          SRTLIS==.-DISGO
    623
    624   031422   000000   032166                       TSTLIS                           ; LIST - list all tests
    625   031423   000000   032216                       TSTLST                           ; LISTEB - list EBus module tests
    626   031424   000000   032216                       TSTLST                           ; LISTSE - list Sequencer related tests
    627   031425   000000   032216                       TSTLST                           ; LISTAL - list 2901's related tests
    628   031426   000000   032216                       TSTLST                           ; LISTMP - list MPROC module tests
    629   031427   000000   032216                       TSTLST                           ; LISTCB - list CBus module tests
    630
    631   031430   000000   032672                       TSTSWI                           ; SWITCH - input switches
    632   031431   000000   032677                       TSTSWP                           ; SWPRIN - print switch selections
    633
    634                        000036          SWSTT==.-DISGO
    635
    636   031432   000000   032704                       TSTSSW                           ; NOPNT
    637   031433   000000   032704                       TSTSSW                           ; PNTLPT
    638   031434   000000   032704                       TSTSSW                           ; DING
    639   031435   000000   032704                       TSTSSW                           ; LOOPER
    640   031436   000000   032704                       TSTSSW                           ; ERSTOP
    641   031437   000000   032704                       TSTSSW                           ; PALERS
    642   031440   000000   032704                       TSTSSW                           ; RELIAB
    643   031441   000000   032704                       TSTSSW                           ; TXTINH
    644   031442   000000   032704                       TSTSSW                           ; OPRSEL
    645   031443   000000   032704                       TSTSSW                           ; TRACE
    646   031444   000000   032704                       TSTSSW                           ; INHFLT
    647   031445   000000   032704                       TSTSSW                           ; INHMSG
    648   031446   000000   032704                       TSTSSW                           ; LOOPGM
    649   031447   000000   032704                       TSTSSW                           ; LOOPTS
    650   031450   000000   032704                       TSTSSW                           ; RUNALL
    651   031451   000000   032704                       TSTSSW                           ; DSPEAR
    652   031452   000000   032704                       TSTSSW                           ; UDEBUG
    653   031453   000000   032704                       TSTSSW                           ; MDEBUG
    654   031454   000000   032704                       TSTSSW                           ; SDEBUG
    655   031455   000000   032704                       TSTSSW                           ; LDEBUG
    656   031456   000000   032704                       TSTSSW                           ; IDEBUG
    657   031457   000000   032704                       TSTSSW                           ; NIPORT
    658   031460   000000   032704                       TSTSSW                           ; CIPORT
    659   031461   000000   032704                       TSTSSW                           ; MMPROC
    660   031462   000000   032704                       TSTSSW                           ; MCBUS
```

```

                                       ;#*********************************************************************
                                       ;* MBUS - Execute MBUS related tests
                                       ;#*********************************************************************

031463  402 00 0 00 064612'            TSTMBU: SETZM   MBARGF#                 ; clear 'MBUS argument given' flag
031464  336 00 0 00 031022*                    SKIPN   ARGFLG                  ; argument given?
031465  254 00 0 00 031476                     JRST    TSTMB0                  ; no - continue
031466  210 01 0 00 052635                     MOVN    1,[FMSGCD (Cmd RepeatCount <CR>)]
031467  260 17 0 00 000000*                    GO      .DARG                   ; yes - get repeat count argument
031470  263 17 0 00 000000                      RTN                            ; error/altmode/question - exit
031471  334 00 0 00 000000                      SKIPA                          ; no argument given
031472  200 00 0 00 000000*                    MOVE    ARGUM                   ; get argument
031473  202 00 0 00 064611'                    MOVEM   MBARG#                  ; save it
031474  402 00 0 00 031464*                    SETZM   ARGFLG                  ; fool TSTEXE argument handling
031475  476 00 0 00 064612'                    SETOM   MBARGF                  ; set 'MBUS argument given' flag

                                       ; Print directions

031476  037 01 0 00 000026             TSTMB0: PFORCE                          ; handle Control-O
                                               TMSGC   <
                                       To isolate a possible MBUS problem:

                                         (1) Disconnect MBUS cable from EBUS module.
                                             Run MBUS tests with MMPROC and MCBUS switches set.
                                             Failed - EBUS module is at fault.  Stop.
                                             Passed - MBUS cable or MPROC or CBUS modules at fault.
031477  037 00 0 00 052636                            Continue at next step to isolate further.>
                                               TMSGC   <
                                         (2) Connect MBUS cable to EBUS module only.
                                             Run MBUS tests with MMPROC and MCBUS switches set.
                                             Failed - MBUS cable is at fault.  Stop.
                                             Passed - MBUS cable or MPROC or CBUS modules at fault.
031500  037 00 0 00 052734                            Continue at next step to isolate further.>
                                               TMSGCD  <
                                         (3) Connect MBUS cable to EBUS and MPROC modules only.
                                             Run MBUS tests with MCBUS switch set.
                                             Failed - MPROC module is at fault.  Stop.
031501  037 00 0 00 053022                   Passed - CBUS module at fault.  Stop.>
031502  037 01 0 00 000026                     PFORCE
031503  200 00 0 00 053072                     MOVE    [MBUS]                  ; get test mask
031504  202 00 0 00 047017                     MOVEM   TSTMSK                  ; save test mask
031505  254 00 0 00 031510                     PJRST   TSTEXG                  ; do the tests
```

```




                                   ;#*********************************************************************
                                   ;* ALL    Execute all tests
                                   ;* EBUS   Execute EBus module tests
                                   ;* SEQ    Execute SEQ related tests
                                   ;* ALU    Execute 2901 related tests
                                   ;* MPROC  Execute MPROC module tests
                                   ;* MBUS   Execute MBUS related tests
                                   ;* CBUS   Execute CBus module tests
                                   ;
                                   ; All of these commands dispatch to TSTEXE which sets up test mask
                                   ; and does dispatch to tests.
                                   ;#*********************************************************************

                                   TSTEXE: MOVE    [ALL                          ; get test mask
                                                    EBUS
                                                    SEQ
                                                    ALU
                                                    MPROC
                                                    CBUS
031506  200 00 0 01 053072                          MBUS]-1(1)
031507  202 00 0 00 047017                 MOVEM   TSTMSK                        ; save test mask
031510  260 17 0 00 033046         TSTEXG: GO      TSTDIS                        ; go dispatch to tests

                                   ; Handle SPEAR reporting

031511  400 15 0 00 000000                 SETZ    ERFLG,                        ; clear error flag
031512  260 17 0 00 044234                 GO      SPREP2                        ; do final SPEAR report

                                   ; Do fault isolation

031513  260 17 0 00 031515                 GO      ISOPNT
031514  263 17 0 00 000000                 RTN
```

```
                                        ;#*******************************************************************
                                        ;* ISOPNT - Print fault isolation data
                                        ;
                                        ; The data printed is that collected by the test dispatch routine
                                        ; consisting of 'NETBAD', the list of bad networks.  Also printed
                                        ; is the module and name associated with each network.
                                        ;#*******************************************************************

031515  332 00 0 00 047022              ISOPNT: SKIPE   TSTMUL                          ; multiple tests being run?
031516  332 00 0 00 047023                      SKIPE   TSTINH                          ; yes - inhibit fault isolation?
031517  263 17 0 00 000000                      RTN                                     ; yes - don't print anything
031520  332 00 0 00 064604'                     SKIPE   FLTPNT#                         ; already printed?
031521  263 17 0 00 000000                      RTN                                     ; yes - return
031522  476 00 0 00 064604'                     SETOM   FLTPNT                          ; ensure it doesn't get printed again
031523  261 17 0 00 000000                      RPUT    (0,1,2,3,4,5)                   ; save AC's
031531  402 00 0 00 031643                      SETZM   NETPNT                          ; first initialize list of
031532  200 00 0 00 053102                      MOVE    [NETPNT,,NETPNT+1]              ;   networks to all zeros
031533  251 00 0 00 031752                      BLT     NETPNT+^D71                     ;   (no networks)
031534  403 01 0 00 000002                      SETZB   1,2                             ; clear AC1 and AC2
031535  350 00 0 00 000002              ISOPN0: AOS     2                               ; point to next network
031536  303 02 0 00 000110                      CAILE   2,^D72                          ; all done?
031537  254 00 0 00 031546                      JRST    ISOPN1                          ; yes - go print it
031540  200 00 0 02 031752                      MOVE    NETBAD-1(2)                     ; get probability of being bad
031541  322 00 0 00 031535                      JUMPE   ISOPN0                          ; zero - check next network
031542  516 00 0 01 031643                      HRLZM   NETPNT(1)                       ; save probability in left half
031543  542 02 0 01 031643                      HRRM    2,NETPNT(1)                     ; save network number in right half
031544  350 00 0 00 000001                      AOS     1                               ; increment number of networks to print
031545  254 00 0 00 031535                      JRST    ISOPN0                          ; check next network
031546  322 01 0 00 031634              ISOPN1: JUMPE   1,ISOPNX                        ; any to print? no - exit

                                        ; Sort the results

031547  400 02 0 00 000000                      SETZ    2,                              ; clear pointer to NETPNT
031550  350 00 0 00 000002              ISOPN2: AOS     2                               ; point to next item
031551  311 02 0 00 000001                      CAML    2,1                             ; done with list?
031552  254 00 0 00 031575                      JRST    ISOPN5                          ; yes - go print it
031553  200 03 0 00 000002                      MOVE    3,2                             ; get pointer to rest of list
031554  350 00 0 00 000003              ISOPN3: AOS     3                               ; point to next item
031555  313 03 0 00 000001                      CAMLE   3,1                             ; done with rest of list?
031556  254 00 0 00 031550                      JRST    ISOPN2                          ; yes - try next item
031557  574 04 0 02 031642                      HLRE    4,NETPNT-1(2)                   ; no - get the probability of
031560  574 05 0 03 031642                      HLRE    5,NETPNT-1(3)                   ;   each network
031561  311 04 0 00 000005                      CAML    4,5                             ; need to reverse items?
031562  254 00 0 00 031554                      JRST    ISOPN3                          ; no - try next item
031563  312 04 0 00 000005                      CAME    4,5                             ; need to reverse items?
031564  254 00 0 00 031571                      JRST    ISOPN4                          ; yes - do it
```

```
031565  550 04 0 02 031642                  HRRZ    4,NETPNT-1(2)           ; maybe - get the network
031566  550 05 0 03 031642                  HRRZ    5,NETPNT-1(3)           ;   numbers of each
031567  311 04 0 00 000005                  CAML    4,5                     ; need to reverse items?
031570  254 00 0 00 031554                  JRST    ISOPN3                  ; no - try next item
031571  200 00 0 02 031642          ISOPN4: MOVE    NETPNT-1(2)             ; yes - do so
031572  250 00 0 03 031642                  EXCH    NETPNT-1(3)
031573  202 00 0 02 031642                  MOVEM   NETPNT-1(2)
031574  254 00 0 00 031554                  JRST    ISOPN3                  ; keep looping

                                    ; Print the results

031575  037 00 0 00 030242          ISOPN5: PCRL
031576  260 17 0 00 000000*                 GO      PNTDEV                  ; print device
031577  550 02 0 00 031643                  HRRZ    2,NETPNT                ; get network offset
031600  200 02 0 02 032063                  MOVE    2,NETDES(2)             ; get netword description
031601  603 02 0 00 400000                  TLNE    2,(1B0)                 ; EBUS module?
031602  037 00 0 00 053103                  TMSGD   < FRU:  EBUS module>    ; yes - print it
031603  603 02 0 00 200000                  TLNE    2,(1B1)                 ; MBUS module?
031604  037 00 0 00 053110                  TMSGD   < FRU:  MPROC module>   ; yes - print it
031605  603 02 0 00 100000                  TLNE    2,(1B2)                 ; CBUS module?
031606  037 00 0 00 053115                  TMSGD   < FRU:  CBUS module>    ; yes - print it
031607  603 02 0 00 040000                  TLNE    2,(1B3)                 ; MBUS cable?
031610  037 00 0 00 053122                  TMSGD   < FRU:  MBUS cable>     ; yes - print it
031611  037 00 0 00 053126                  TMSGCD  <Faulty networks (in order of probability):>
031612  474 03 0 00 000000                  SETO    3,                      ; pointer to list of networks
031613  350 00 0 00 000003          ISOPN6: AOS     3                       ; point to next network
031614  200 02 0 03 031643                  MOVE    2,NETPNT(3)             ; get data
031615  322 02 0 00 031633                  JUMPE   2,ISOPN8                ; done? yes - exit
031616  550 02 0 00 000002                  HRRZ    2,2                     ; get only network offset
031617  550 02 0 02 032062                  HRRZ    2,NETDES-1(2)           ; get address of ASCIZ message
031620  037 00 0 00 053140                  TMSGC   <  >
031621  336 00 0 00 064606'                 SKIPN   IDEBUG                  ; fault isolation switch set?
031622  254 00 0 00 031627                  JRST    ISOPN7                  ; no - continue
031623  037 00 0 00 053141                  TMSG    <Weight >               ; yes - print the weight
031624  574 00 0 03 031643                  HLRE    NETPNT(3)               ; get data
031625  037 16 0 00 000003                  PNTOCS
031626  037 00 0 00 000040                  PSP
031627  037 00 1 00 000002          ISOPN7: PNTMSG  @2                      ; print it
031630  336 00 0 00 064606'                 SKIPN   IDEBUG                  ; fault isolation switch set?
031631  307 03 0 00 000003                  CAIG    3,3                     ; no - already printed 4 networks?
031632  254 00 0 00 031613                  JRST    ISOPN6                  ; no - loop till done

                                    ; Exit

031633  037 00 0 00 030242          ISOPN8: PCRL                            ; final CRLF
031634  262 17 0 00 000005          ISOPNX: RGET    (5,4,3,2,1,0)           ; restore AC's
031642  263 17 0 00 000000                  RTN                             ; return
```

```
                                 ;#**************************************************************************
                                 ;* Network Data Areas
                                 ;#**************************************************************************

031643                           NETPNT: BLOCK   ^D72                        ; networks in order of probability
031753                           NETBAD: BLOCK   ^D72                        ; probability that Network n is bad

        400000  000000           NEBUS==1B0
        200000  000000           NMPROC==1B1
        100000  000000           NCBUS==1B2
        040000  000000           NMBUS==1B3

                                 ; EBUS Module networks

032063  400000  053143           NETDES: NEBUS![ASCIZ ^E1:  EBUS Module - EBUS Control Logic^]
032064  400000  053153                   NEBUS![ASCIZ ^E2:  EBUS Module - CC EBUS RQST^]
032065  400000  053162                   NEBUS![ASCIZ ^E3:  EBUS Module - CC GRNT CSR^]
032066  400000  053171                   NEBUS![ASCIZ ^E4:  EBUS Module - Interrupt Logic^]
032067  400000  053200                   NEBUS![ASCIZ ^E5:  EBUS Module - Clock Logic^]
032070  400000  053207                   NEBUS![ASCIZ ^E6:  EBUS Module - Clear Logic^]
032071  400000  053216                   NEBUS![ASCIZ ^E7:  EBUS Module - Magic # Field Decode^]
032072  400000  053226                   NEBUS![ASCIZ ^E8:  EBUS Module - CC CSR CHNG^]
032073  400000  053235                   NEBUS![ASCIZ ^E9:  EBUS Module - KMUX^]
032074  400000  053242                   NEBUS![ASCIZ ^E10: EBUS Module - EMUX^]
032075  400000  053247                   NEBUS![ASCIZ ^E11: EBUS Module - RMUX/CSR 25-31^]
032076  400000  053256                   NEBUS![ASCIZ ^E12: EBUS Module - CSR 11-13/19-22/33-35^]
032077  400000  053267                   NEBUS![ASCIZ ^E13: EBUS Module - CSR 04 Rqst Exam/Dep^]
032100  400000  053277                   NEBUS![ASCIZ ^E14: EBUS Module - CSR 05 Rqst Interrupt/CC INTR ACTIVE^]
032101  400000  053313                   NEBUS![ASCIZ ^E15: EBUS Module - CSR 06/07 CRAM PE/MBUS Error^]
032102  400000  053325                   NEBUS![ASCIZ ^E16: EBUS Module - CSR 24 EBUS PE^]
032103  400000  053334                   NEBUS![ASCIZ ^E17: EBUS Module - CSR 32 Mproc Run^]
032104  400000  053344                   NEBUS![ASCIZ ^E18: EBUS Module - KMUX Parity Generation^]
032105  400000  053355                   NEBUS![ASCIZ ^E19: EBUS Module - EBUS Parity Checking^]
032106  400000  053365                   NEBUS![ASCIZ ^E20: EBUS Module - EBUF (Check MBUS cable connection)^]
032107  400000  053400                   NEBUS![ASCIZ ^E21: EBUS Module - EBUS Data Xcvr's^]
032110  400000  053410                   NEBUS![ASCIZ ^E22: EBUS Module - EBUS Control Xcvr's^]
032111  400000  053420                   NEBUS![ASCIZ ^E23: EBUS Module - 2901's/Carry Logic^]
032112  400000  053430                   NEBUS![ASCIZ ^E24: EBUS Module - Constant MUX^]

                                 ; MPROC Module networks

032113  200000  053437                   NMPROC![ASCIZ ^M1:  MPROC Module - MPROC Run Clocks^]
032114  200000  053447                   NMPROC![ASCIZ ^M2:  MPROC Module - Local Memory Control^]
032115  200000  053460                   NMPROC![ASCIZ ^M3:  MPROC Module - MBUS Error Detect Logic^]
032116  200000  053471                   NMPROC![ASCIZ ^M4:  MPROC Module - CRAM Read/Write Control^]
032117  200000  053502                   NMPROC![ASCIZ ^M5:  MPROC Module - Jump ADDR MUX^]
032120  200000  053511                   NMPROC![ASCIZ ^M6:  MPROC Module - CC MUX^]
032121  200000  053517                   NMPROC![ASCIZ ^M7:  MPROC Module - 2910 MicroSequencer^]
032122  200000  053527                   NMPROC![ASCIZ ^M8:  MPROC Module - CRAM ADDR MUX^]
```

```
032123  200000  053536                  NMPROC![ASCIZ ^M9:  MPROC Module - LAR^]
032124  200000  053543                  NMPROC![ASCIZ ^M10: MPROC Module - CRAM 0000-7777^]
032125  200000  053552                  NMPROC![ASCIZ ^M11: MPROC Module - Cond/Skip Decoder^]
032126  200000  053562                  NMPROC![ASCIZ ^M12: MPROC Module - CRAM Control Register^]
032127  200000  053573                  NMPROC![ASCIZ ^M13: MPROC Module - Local Storage RAM^]
032130  200000  053603                  NMPROC![ASCIZ ^M14: MPROC Module - Local Storage Addr Register^]
032131  200000  053615                  NMPROC![ASCIZ ^M15: MPROC Module - RAM Mode MUX^]
032132  200000  053624                  NMPROC![ASCIZ ^M16: MPROC Module - RAR^]
032133  200000  053631                  NMPROC![ASCIZ ^M17: MPROC Module - Right CRAM Load Buffer^]
032134  200000  053642                  NMPROC![ASCIZ ^M18: MPROC Module - Left CRAM Load Buffer^]
032135  200000  053653                  NMPROC![ASCIZ ^M19: MPROC Module - MW Out MUX^]

                                ; CBUS Module networks

032136  100000  053662                  NCBUS![ASCIZ ^C1:  CBUS Module - CBUS Control Logic^]
032137  100000  053672                  NCBUS![ASCIZ ^C2:  CBUS Module - Clock Logic^]
032140  100000  053701                  NCBUS![ASCIZ ^C3:  CBUS Module - Reset Logic^]
032141  100000  053710                  NCBUS![ASCIZ ^C4:  CBUS Module - Clear Logic^]
032142  100000  053717                  NCBUS![ASCIZ ^C5:  CBUS Module - 'CC CBUS AVAIL'^]
032143  100000  053726                  NCBUS![ASCIZ ^C6:  CBUS Module - 'CC CB LST WD'/ 'CC CHAN ERR'^]
032144  100000  053740                  NCBUS![ASCIZ ^C7:  CBUS Module - 'CLR CC CODE'^]
032145  100000  053747                  NCBUS![ASCIZ ^C8:  CBUS Module - Magic # Field Decode^]
032146  100000  053757                  NCBUS![ASCIZ ^C9:  CBUS Module - 'CC CBUS PAR ERR'^]
032147  100000  053767                  NCBUS![ASCIZ ^C10: CBUS Module - 'CC PLI PAR ERR'^]
032150  100000  053777                  NCBUS![ASCIZ ^C11: CBUS Module - Parity Predictor^]
032151  100000  054007                  NCBUS![ASCIZ ^C12: CBUS Module - Formatter Control^]
032152  100000  054017                  NCBUS![ASCIZ ^C13: CBUS Module - PLI Control Logic^]
032153  100000  054027                  NCBUS![ASCIZ ^C14: CBUS Module - CBUS Input Buffer^]
032154  100000  054037                  NCBUS![ASCIZ ^C15: CBUS Module - CBUS Output Buffer^]
032155  100000  054047                  NCBUS![ASCIZ ^C16: CBUS Module - CBUS Out Par Gen^]
032156  100000  054057                  NCBUS![ASCIZ ^C17: CBUS Module - CBUS In Par Checker^]
032157  100000  054067                  NCBUS![ASCIZ ^C18: CBUS Module - DMUX^]
032160  100000  054074                  NCBUS![ASCIZ ^C19: CBUS Module - SMUX^]
032161  100000  054101                  NCBUS![ASCIZ ^C20: CBUS Module - Mvr/Fmtr^]
032162  100000  054107                  NCBUS![ASCIZ ^C21: CBUS Module - PLI Logic (Check PLI cable connection)^]
032163  100000  054123                  NCBUS![ASCIZ ^C22: CBUS Module - PMUX^]
032164  100000  054130                  NCBUS![ASCIZ ^C23: CBUS Module - CMUX^]
032165  100000  054135                  NCBUS![ASCIZ ^C24: CBUS Module - CBUF^]
```

```
                                    ;#*********************************************************************
                                    ;* TSTLIS - List test description for all tests.
                                    ;
                                    ; Argument:  XXn - list only one test
                                    ;            none - list all tests
                                    ;#*********************************************************************

                                    ; Check for argument

032166  201 00 0 00 001000          TSTLIS: MOVEI   1000                    ; set up default maximum
032167  202 00 0 00 064651'                 MOVEM   TSTL2                   ;   test limit
032170  476 00 0 00 047023                  SETOM   TSTINH                  ; inhibit fault isolation
032171  336 00 0 00 031474*                 SKIPN   ARGFLG                  ; argument given?
032172  254 00 0 00 032210                  JRST    TSTLI1                  ; no - do all tests

                                    ; Decode argument

032173  200 01 0 00 054153          TSTLI0: MOVE    1,[FMSGCD (LIST <CR> or LIST XXnnn XXnnn ... <CR>)]
032174  260 17 0 00 000000*                 GO      .SARG                   ; get SIXBIT argument
032175  263 17 0 00 000000                   RTN                            ; error/altmode/question - exit
032176  263 17 0 00 000000                   RTN                            ; no more arguments
032177  200 01 0 00 031472*                 MOVE    1,ARGUM                 ; get argument
032200  260 17 0 00 034315                  GO      TSTDEC                  ; decode as a test name
                                             JRST   [FMSGCD <? Argument not recognizable as a test name 'XXn'>
                                                     GO      FIOFF
032201  254 00 0 00 054167                           RTN]
                                             JRST   [FMSGC  <? Test selected is not in range 1->
                                                     MOVE    1              ; get range
                                                     PNTOCF                 ; print it
                                                     PCRLF
                                                     GO      FIOFF
032202  254 00 0 00 054202                           RTN]
032203  200 00 0 00 047015                  MOVE    TSTADD                  ; get dispatch table address
032204  202 00 0 00 064650'                 MOVEM   TSTL1#                  ; save it as start address
032205  202 00 0 00 064651'                 MOVEM   TSTL2#                  ; save it as end address
032206  260 17 0 00 032250                  GO      TSTLSS                  ; print it
032207  254 00 0 00 032173                  JRST    TSTLI0                  ; next argument

                                    ; List all tests

032210  201 00 0 00 033330          TSTLI1: MOVEI   TDISP1                  ; get dispatch table address
032211  202 00 0 00 064650'                 MOVEM   TSTL1                   ; save it as start address
032212  201 00 0 00 033700                  MOVEI   TEND-1                  ; get end of table
032213  202 00 0 00 064651'                 MOVEM   TSTL2                   ; save it as end address
032214  260 17 0 00 032250                  GO      TSTLSS                  ; print them
032215  263 17 0 00 000000                  RTN                             ; yes - return
```

```
                                        ;#*********************************************************************
                                        ;* TSTLST - List test description for either single test or class of tests.
                                        ;
                                        ; Argument(s):  n - list only one test
                                        ;               n,n - list range of tests
                                        ;               none - list all tests
                                        ;#*********************************************************************

                                        ; First set up class of tests

032216  275 01 0 00 000027              TSTLST: SUBI    1,SRTLIS+1              ; normalize to 0..n
032217  202 01 0 00 047010                      MOVEM   1,TSTCLS                ; save test class
                                                XCT     [MOVEI  2,TEBUS         ; get starting address in address
                                                         MOVEI  2,TSEQ          ;   table
                                                         MOVEI  2,TALU
                                                         MOVEI  2,TMPROC
032220  256 00 0 01 054210                               MOVEI  2,TCBUS](1)
032221  202 02 0 00 064650'                     MOVEM   2,TSTL1                 ; save as starting address
032222  202 02 0 00 064652'                     MOVEM   2,TSTL3#                ; save for later use
                                                XCT     [MOVEI  2,TSEQ-1        ; get starting address in address
                                                         MOVEI  2,TALU-1        ;   table
                                                         MOVEI  2,TMPROC-1
                                                         MOVEI  2,TCBUS-1
032223  256 00 0 01 054215                               MOVEI  2,TEND-1](1)
032224  202 02 0 00 064651'                     MOVEM   2,TSTL2                 ; save as starting address
032225  200 03 0 00 000002                      MOVE    3,2                     ; calculate maximum test number
032226  274 03 0 00 064650'                     SUB     3,TSTL1                 ;   that can be specified

                                        ; Check for argument(s)

032227  200 01 0 00 054234                      MOVE    1,[FMSGCD <LISTxx test# (CR) or LISTxx test#,test# (CR)>]
032230  260 17 0 00 031467*                     GO      .DARG                   ; get test number
032231  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
032232  254 00 0 00 032250                        JRST  TSTLSS                  ; no argument given
032233  370 01 0 00 032177*                     SOS     1,ARGUM                 ; get argument (normalized to 0..n)
032234  311 01 0 00 000003                      CAML    1,3                     ; out of range?
                                                JRST    [FMSGC  <? Test number out of range 1 - >
                                                         MOVE   3
                                                         PNTOCF
                                                         PCRLF
                                                         GO     FIOFF
032235  254 00 0 00 054244                               RTN]
032236  272 01 0 00 064650'                     ADDM    1,TSTL1                 ; adjust first test number
```

```
032237  210 01 0 00 054234           MOVN    1,[FMSGCD <LISTxx test# (CR) or LISTxx test#,test# (CR)>]
032240  260 17 0 00 032230*          GO      .DARG                   ; get test number
032241  263 17 0 00 000000            RTN                            ; error/altmode/question - exit
                                      JRST   [MOVE   TSTL1           ; no 2nd argument given
                                              MOVEM  TSTL2
032242  254 00 0 00 054252                    JRST   TSTLSS]
032243  370 01 0 00 032233*          SOS     1,ARGUM                 ; get argument (normalized to 0..n)
032244  311 01 0 00 000003           CAML    1,3                     ; out of range?
                                     JRST    [FMSGC  <? Test number out of range 1 - >
                                              MOVE   3
                                              PNTOCF
                                              PCRLF
                                              GO     FIOFF
032245  254 00 0 00 054244                    RTN]
032246  270 01 0 00 064652'          ADD     1,TSTL3                 ; add to base
032247  202 01 0 00 064651'          MOVEM   1,TSTL2                 ; save second test number

                             ; Print the test descriptions

032250  200 01 0 00 064650'  TSTLSS: MOVE    1,TSTL1                 ; get dispatch table address
032251  037 00 0 00 030242           PCRL
032252  313 01 0 00 064651'          CAMLE   1,TSTL2                 ; done yet?
032253  263 17 0 00 000000           RTN                             ; yes - return
032254  202 01 0 00 047015           MOVEM   1,TSTADD                ; save dispatch table address
032255  260 17 0 00 042775           GO      TSTGET                  ; get test data
032256  260 17 0 00 043076           GO      TSTPNT                  ; no - print it
032257  350 00 0 00 064650'          AOS     TSTL1                   ; point to next test
032260  037 07 0 00 000003           TTALTM                          ; no - altmode typed?
032261  254 00 0 00 032250           JRST    TSTLSS                  ; no - keep loopint
032262  037 00 0 00 030242           PCRL                            ; yes - a final CRLF
032263  476 00 0 00 064570'          SETOM   ALTF                    ; set altmode flag
032264  263 17 0 00 000000           RTN                             ; return
```

```
                                         ;#*********************************************************************
                                         ;* TSTDIA - Return to DIAMON
                                         ;#*********************************************************************

032265  200 01 0 00 054261              TSTDIA: MOVE    1,[FMSGCD (DIAMON <CR>)]
032266  260 17 0 00 000000*                     GO      CHKARG                  ; check for argument
032267  263 17 0 00 000000                      RTN                             ; error - exit
032270  260 17 0 00 030727*                     GO      DEVREL                  ; release the device
032271  254 00 0 00 020000                      JRST    20000                   ; enter DIAMON


                                         ;#*********************************************************************
                                         ;* TSTDDT - Enter DDT
                                         ;#*********************************************************************

032272  200 01 0 00 054265              TSTDDT: MOVE    1,[FMSGCD (DDT <CR>)]
032273  260 17 0 00 032266*                     GO      CHKARG                  ; check for argument
032274  263 17 0 00 000000                      RTN                             ; error - exit
032275  037 01 0 00 054266                      FMSGCD  <Type RTN$X to continue>
032276  260 17 0 00 030007                      GO      SRTDDT                  ; enter DDT
032277  260 17 0 00 000000*                     GO      INITPD                  ; init PI system
032300  260 17 0 00 030776*                     GO      SETVEC                  ; set up vector addresses
032301  400 01 0 00 000000                      SETZ    1,                      ; clear AC1
032302  263 17 0 00 000000                      RTN                             ; ask for a test


                                         ;#*********************************************************************
                                         ;* TSTHLT - Program halts - CONTINUE will start it back up.
                                         ;#*********************************************************************

032303  200 01 0 00 054277              TSTHLT: MOVE    1,[FMSGCD (EXIT <CR>)]
032304  260 17 0 00 032273*                     GO      CHKARG                  ; check for argument
032305  263 17 0 00 000000                      RTN                             ; error - exit
032306  260 17 0 00 032270*                     GO      DEVREL                  ; release the device
032307  260 17 0 00 000000*                     GO      .CLOSE                  ; close any open files
032310  332 00 0 00 020000                      SKIPE   20000                   ; DIAMON loaded?
032311  254 00 0 00 020000                      JRST    20000                   ; yes - return there
032312  254 04 0 00 032313                      HALT    .+1                     ; no - halt
032313  254 00 0 00 030610                      JRST    START                   ; continue
```

```
                                     ;#***********************************************************************
                                     ;* CONFIG - Determine/print device configuration
                                     ;
                                     ; Select word (NISEL or CISEL):
                                     ;  Bit 35 - Port exists - 0 - no    (This is determined by whether or not
                                     ;                         1 - yes    a CONI succeeds and bit 0 is set,
                                     ;                                    indicating a port).
                                     ;      34 - Port selectable - 0 - no   (This is determined by whether
                                     ;                             1 - yes   or not a DEVREQ succeeds).
                                     ;      33 - Port selected - 0 - no
                                     ;                           1 - yes
                                     ;#***********************************************************************

                                     ; Determine which ports are available and obtain each from system

032314  261 17 0 00 000000           CONFIG: RPUT    (0,1,2,3,4)             ; save AC's
032321  260 17 0 00 000000*                  GO      DSETUP                  ; set up DIAG JSYS
032322  260 17 0 00 000000*                  GO      USRION                  ; set USRIOT bit
032323  201 03 0 00 000003                   MOVEI   3,3                     ; select NI port
032324  201 04 0 00 000003                   MOVEI   4,3                     ; select CI port
032325  336 00 0 00 064654'                  SKIPN   UDEBUG                  ; debug mode?
032326  403 03 0 00 000004                   SETZB   3,4                     ; no - clear NI and CI selection
032327  260 17 0 00 031070*                  GO      SWITT                   ; Get the switches
032330  200 02 0 00 000000                   MOVE    2,0                     ; Copy them to AC2
032331  606 02 0 00 000060                   TRNN    2,CIPORT!NIPORT         ; Neither set?
032332  660 02 0 00 000060                   TRO     2,CIPORT!NIPORT         ; Yes, try for both of them

                                     ; Check for NI port

032333  200 16 0 00 047001                   MOVE    MBCN,PORTNI             ; get NI port number
032334  260 17 0 00 000000*                  GO      RDCSR                   ; read CSR register
032335  400 01 0 00 000000                    SETZ   1,                      ; failed
032336  603 01 0 00 400000                   TLNE    1,400000                ; is this a port?
032337  660 03 0 00 000001                   TRO     3,1                     ; yes - indicate such

                                     ; Check for CI port

032340  200 16 0 00 047002                   MOVE    MBCN,PORTCI             ; get CI port number
032341  260 17 0 00 032334*                  GO      RDCSR                   ; read CSR register
032342  400 01 0 00 000000                    SETZ   1,                      ; failed
032343  603 01 0 00 400000                   TLNE    1,400000                ; is this a port?
032344  660 04 0 00 000001                   TRO     4,1                     ; yes - indicate such
```

```
                                        ; Now check to see if NI port is available

032345  200 16 0 00 [illegible]001               MOVE    MBCN,PORTNI                   ; set up MBCN number
032346  602 02 0 00 000040               TRNE    2,NIPORT                      ; Looking for the NI port today?
032347  260 17 0 00 031060*              GO      DEVREQ                        ; request the device
032350  334 00 0 00 000000                SKIPA                                ; Failed or not looking - continue
032351  660 03 0 00 000002               TRO     3,2                           ; set available bit

                                        ; Now check to see if CI port is available

032352  200 16 0 00 047002               MOVE    MBCN,PORTCI                   ; set up MBCN number
032353  602 02 0 00 000020               TRNE    2,CIPORT                      ; Looking for CI port today?
032354  260 17 0 00 032347*              GO      DEVREQ                        ; request the device
032355  334 00 0 00 000000                SKIPA                                ; Failed or not looking - continue
032356  660 04 0 00 000002               TRO     4,2                           ; set available bit

                                        ; Now look at select switches and determine selection

032357  602 02 0 00 000040               TRNE    2,NIPORT                      ; NI port selected?
032360  302 03 0 00 000003               CAIE    3,3                           ; Yes, port exists and selectable?
032361  304 00 0 00 000000               CAIA                                  ; No to either
032362  660 03 0 00 000004               TRO     3,4                           ; NIA20 exists, selectable, selected
032363  202 03 0 00 047003               MOVEM   3,NISEL                       ; save NI select bits

032364  602 02 0 00 000020               TRNE    2,CIPORT                      ; CI port selected?
032365  302 04 0 00 000003               CAIE    4,3                           ; Yes, port exists and is selectable?
032366  304 00 0 00 000000               CAIA                                  ; No to either
032367  660 04 0 00 000004               TRO     4,4                           ; CI20 exists, selectable, selected
032370  202 04 0 00 047004               MOVEM   4,CISEL                       ; save CI select bits

                                        ; Print configuration

032371  260 17 0 00 032400               GO      CONPNT                        ; print configuration noted
032372  262 17 0 00 000004               RGET    (4,3,2,1,0)                   ; restore AC's
032377  263 17 0 00 000000               RTN                                   ; return
```

```

                                            ;#**********************************************************************
                                            ;* CONPNT - Print configuration
                                            ;#    *******************************************************************

032400   261 17 0 00 000000                 CONPNT: RPUT    (0,1)                           ; save AC's
032402   037 00 0 00 054300                         TMSGC   <Port Configuration:>
032403   037 00 0 00 054305                         TMSGC   <  Port  Exists  Selectable  Selected>
032404   037 00 0 00 054315                         TMSGC   <   NI      >
032405   200 01 0 00 047003                         MOVE    1,NISEL
                                                    XCT     [TMSG    <No        No          No>
                                                             TMSG    <Yes       No          No>
                                                             TMSG    <No        Yes         No>
                                                             TMSG    <Yes       Yes         No>
                                                             TMSG    <No        No          No>
                                                             TMSG    <Yes       No          No>
                                                             TMSG    <No        Yes         Yes>
032406   256 00 0 01 054360                                  TMSG    <Yes       Yes         Yes>](1)
032407   037 00 0 00 054370                         TMSGC   <   CI      >
032410   200 01 0 00 047004                         MOVE    1,CISEL
                                                    XCT     [TMSG    <No        No          No>
                                                             TMSG    <Yes       No          No>
                                                             TMSG    <No        Yes         No>
                                                             TMSG    <Yes       Yes         No>
                                                             TMSG    <No        No          No>
                                                             TMSG    <Yes       No          No>
                                                             TMSG    <No        Yes         Yes>
032411   256 00 0 01 054360                                  TMSG    <Yes       Yes         Yes>](1)
032412   037 00 0 00 030242                         PCRL
032413   262 17 0 00 000001                         RGET    (1,0)                           ; restore AC's
032415   263 17 0 00 000000                         RTN                                     ; return
```

```
                                        ;#*********************************************************************
                                        ;* TSTSEL - Select device(s) to test
                                        ;#*********************************************************************

 032416  200 01 0 00 054404             TSTSEL: MOVE    1,[FMSGCD <SELECT NI or NI,CI or CI or SELECT (CR)>]
 032417  260 17 0 00 032174*                    GO      .SARG                   ; get SIXBIT argument
 032420  263 17 0 00 000000                      RTN                            ; error/altmode/question - exit
 032421  254 00 0 00 032457                      JRST   TSTSEG                  ; no argument - query user
 032422  200 01 0 00 032243*            TSTSE0: MOVE    1,ARGUM                 ; get sixbit command
                                                MOVEI   2,[SIXBIT /NI/          ; get address of cmd list
                                                           SIXBIT /CI/
 032423  201 02 0 00 054405                                0]
 032424  260 17 0 00 031053*                    GO      .COMM                   ; handle command decoding
                                                 JRST   [FMSGCD <? Unrecognizable - Type ? for help>
                                                          GO      FIOFF
 032425  254 00 0 00 054410                               RTN]                  ; error return

                                        ; Have a command

 032426  400 16 0 00 000000                     SETZ    MBCN,                   ; clear current selection
 032427  200 00 0 00 047003                     MOVE    NISEL                   ; get NI select word
 032430  620 00 0 00 000004                     TRZ     4                       ; clear select bit
 032431  202 00 0 00 047003                     MOVEM   NISEL                   ; save it
 032432  200 00 0 00 047004                     MOVE    CISEL                   ; get CI select word
 032433  620 00 0 00 000004                     TRZ     4                       ; clear select bit
 032434  202 00 0 00 047004                     MOVEM   CISEL                   ; save it
 032435  306 01 0 00 000000                     CAIN    1,0                     ; NI specified?
                                                GO      [MOVE   NISEL           ; yes - set the select bit
                                                          TRO    4
                                                          TRNN   2
                                                          JRST   [FMSGCD <? NI port is not selectable>
                                                                   RTN]
                                                          MOVEM  NISEL
 032436  260 17 0 00 054424                               RTN]
 032437  306 01 0 00 000001                     CAIN    1,1                     ; CI specified?
                                                GO      [MOVE   CISEL           ; yes - set the select bit
                                                          TRO    4
                                                          TRNN   2
                                                          JRST   [FMSGCD <? CI port is not selectable>
                                                                   RTN]
                                                          MOVEM  CISEL
 032440  260 17 0 00 054443                               RTN]
```

```

                                    ; Check for another argument

032441  210 01 0 00 054404                  MOVN    1,[FMSGCD <SELECT NI or NI,CI or CI or SELECT (CR)>]
032442  260 17 0 00 032417*                 GO      .SARG                   ; get SIXBIT argument
032443  263 17 0 00 000000                   RTN                            ; error/altmode/question - exit
032444  254 00 0 00 032504                   JRST   TSTSEX                  ; no argument - exit
032445  200 01 0 00 032422*                 MOVE    1,ARGUM                 ; get sixbit command
                                            MOVEI   2,[SIXBIT /NI/          ; get address of cmd list
                                                       SIXBIT /CI/
032446  201 02 0 00 054405                             0]
032447  260 17 0 00 032424*                 GO      .COMM                   ; handle command decoding
                                             JRST   [FMSGCD <? Unrecognizable - Type ? for help>
                                                     GO      FIOFF
032450  254 00 0 00 054410                           RTN]                   ; error return

                                    ; Have a command

032451  400 16 0 00 000000                  SETZ    MBCN,                   ; clear current selection
032452  306 01 0 00 000000                  CAIN    1,0                     ; NI specified?
                                            GO      [MOVE   NISEL           ; yes - set the select bit
                                                     TRO    4
                                                     TRNN   2
                                                     JRST   [FMSGCD <? NI port is not selectable>
                                                             RTN]
                                                     MOVEM  NISEL
032453  260 17 0 00 054424                           RTN]
032454  306 01 0 00 000001                  CAIN    1,1                     ; CI specified?
                                            GO      [MOVE   CISEL           ; yes - set the select bit
                                                     TRO    4
                                                     TRNN   2
                                                     JRST   [FMSGCD <? CI port is not selectable>
                                                             RTN]
                                                     MOVEM  CISEL
032455  260 17 0 00 054443                           RTN]
032456  254 00 0 00 032504                  JRST    TSTSEX                  ; exit

                                    ; Check NI port

032457  400 16 0 00 000000          TSTSEG: SETZ    MBCN,                   ; clear current selection
032460  200 01 0 00 047003                  MOVE    1,NISEL                 ; get NI select word
032461  606 01 0 00 000002                  TRNN    1,2                     ; port selectable?
032462  254 00 0 00 032470                  JRST    TSTSGO                  ; no - continue
032463  200 00 0 00 054455                  MOVE    [FMSGC  <Test NI port?>]
032464  260 17 0 00 000000*                 GO      TTYYES                  ; check for Y or N
032465  624 01 0 00 000004                  TRZA    1,4                     ; no - set no
032466  660 01 0 00 000004                  TRO     1,4                     ; yes - set yes
032467  202 01 0 00 047003                  MOVEM   1,NISEL                 ; save selection
```

```


                                          ; Check CI port

032470  200 01 0 00 047004                TSTSGO: MOVE    1,CISEL                         ; get CI select word
032471  606 01 0 00 000002                        TRNN    1,2                             ; port selectable?
032472  254 00 0 00 032504                        JRST    TSTSEX                          ; no - exit
032473  200 00 0 00 054462                        MOVE    [FMSGC  <Test CI port?>]
032474  260 17 0 00 032464*                       GO      TTYYES                          ; check for Y or N
032475  624 01 0 00 000004                        TRZA    1,4                             ; no - set no
032476  660 01 0 00 000004                        TRO     1,4                             ; yes - set yes
032477  202 01 0 00 047004                        MOVEM   1,CISEL                         ; save selection

                                          ; Set up MBCN properly

032500  200 16 0 00 047001                        MOVE    MBCN,PORTNI                     ; get NI device code
032501  200 01 0 00 047003                        MOVE    1,NISEL                         ; get NI select word
032502  606 01 0 00 000004                        TRNN    1,4                             ; selected?
032503  200 16 0 00 047002                        MOVE    MBCN,PORTCI                     ; no - get CI device code

                                          ; Print selection

032504  200 00 0 00 047003                TSTSEX: MOVE    NISEL                           ; get NI select word
032505  602 00 0 00 000004                        TRNE    4                               ; selected?
032506  037 00 0 00 054463                        TMSGC   <[NI port selected]>            ; yes - print such
032507  200 00 0 00 047004                        MOVE    CISEL                           ; get CI select word
032510  602 00 0 00 000004                          NE    4                               ; selected?
032511  037 00 0 00 054470                          GC    <[CI port selected]>            ; yes - print such
032512  200 00 0 00 047003                                NISEL                           ; get NI select word
032513  434 00 0 00 047004                        IO      ISEL                            ; include CI select word
032514  606 00 0 00 000004                        TRNN                                    ; any selected?
032515  037 00 0 00 054475                        TMSGC           orts selected]>         ; no - print such
032516  037 00 0 00 030242                        PCRL
032517  263 17 0 00 000000                        RTN                                     ; return
```

```
                                              ;#*********************************************************************
                                              ; PTEST - Determine port under test and set up MBCN
                                              ; PTESTP - Print port under test (if not already known)
                                              ;*********************************************************************

032520  261 17 0 00 000000      PTEST:  PUT     0                               ; save AC0
032521  316 16 0 00 047001              CAMN    MBCN,PORTNI                     ; NI port already selected?
032522  254 00 0 00 032535              JRST    PTEST0                          ; yes - continue
032523  316 16 0 00 047002              CAMN    MBCN,PORTCI                     ; no - CI port already selected?
032524  254 00 0 00 032535              JRST    PTEST0                          ; yes - continue
032525  200 16 0 00 047001              MOVE    MBCN,PORTNI                     ; no - set to NI port
032526  200 00 0 00 047003              MOVE    NISEL                           ; get NI select bits
032527  602 00 0 00 000004              TRNE    4                               ; select possible?
032530  254 00 0 00 032535              JRST    PTEST0                          ; yes - continue
032531  200 16 0 00 047002              MOVE    MBCN,PORTCI                     ; no - set to CI port
032532  200 00 0 00 047004              MOVE    CISEL                           ; get CI select bits
032533  606 00 0 00 000004              TRNN    4                               ; select possible?
032534  400 16 0 00 000000              SETZ    MBCN,                           ; no - indicate such
032535  262 17 0 00 000000      PTEST0: GET     0                               ; restore AC0
032536  263 17 0 00 000000              RTN                                     ; return

032537  261 17 0 00 000000      PTESTP: PUT     0                               ; save AC0
032540  200 00 0 00 047003              MOVE    NISEL                           ; get NI select word
032541  404 00 0 00 047004              AND     CISEL                           ;   and CI select word
032542  606 00 0 00 000004              TRNN    4                               ; both selected?
032543  254 00 0 00 032550              JRST    PTESTX                          ; no - continue
032544  316 16 0 00 047001              CAMN    MBCN,PORTNI                     ; NI port already selected?
032545  037 00 0 00 054502              TMSGCD  <[Port under test - NI]>; yes - print such
032546  316 16 0 00 047002              CAMN    MBCN,PORTCI                     ; CI port already selected?
032547  037 00 0 00 054510              TMSGCD  <[Port under test - CI]>; yes - print such
032550  336 00 0 00 000016      PTESTX: SKIPN   MBCN                            ; any port selected?
032551  037 00 0 00 054516              TMSGCD  <[Port under test - none]> ; no - print such
032552  262 17 0 00 000000              GET     0                               ; restore AC0
032553  263 17 0 00 000000              RTN                                     ; exit
```

```

                                ;#***********************************************************************
                                ;* TSTHLP - Help for 'WHAT TEST' question.
                                ;#***********************************************************************

                                TSTHLP: MOVN    1,[FMSGD <
                                HELP ALL     CONFIG
                                     TESTS   LIST
032554  210 01 0 00 054541           MISC    SWITCH>]
032555  260 17 0 00 032442*             GO     .SARG                      ; get SIXBIT argument
032556  263 17 0 00 000000               RTN                               ; error/altmode/question - exit
                                         GO     [MOVE   [SIXBIT /ALL/]    ; no argument - assume all
                                                 MOVEM  ARGUM
032557  260 17 0 00 054543                       RTN]
032560  200 01 0 00 032445*             MOVE    1,ARGUM                   ; get sixbit command
032561  201 02 0 00 032574              MOVEI   2,TSTHLC                  ; get address of cmd list
032562  260 17 0 00 032447*             GO      .COMM                     ; handle command decoding
                                         JRST   [FMSGCD <? Unrecognizable - Type ? for help>
                                                 GO     FIOFF
032563  254 00 0 00 054410                       RTN]                      ; error return
032564  037 01 0 00 000026              PFORCE                            ; handle Control-O
032565  200 02 0 00 000001              MOVE    2,1                       ; get AC1
032566  336 03 0 01 032603      TSTHLO: SKIPN   3,TSTHLL(1)               ; get address of TMSG
032567  263 17 0 00 000000              RTN                               ; all done - exit
032570  256 00 0 00 000003              XCT     3                         ; print it
032571  350 00 0 00 000001              AOS     1                         ; point to next address
032572  322 02 0 00 032566              JUMPE   2,TSTHLO                  ; loop till done
032573  263 17 0 00 000000              RTN                               ; exit

032574  41 54 54 00 00 00       TSTHLC: SIXBIT  /ALL/                     ; print all help info
032575  64 45 63 64 63 00               SIXBIT  /TESTS/                   ; print test commands
032576  55 51 63 43 00 00               SIXBIT  /MISC/                    ; print miscellaneous commands
032577  43 57 56 46 51 47               SIXBIT  /CONFIG/                  ; print configuration commands
032600  54 51 63 64 00 00               SIXBIT  /LIST/                    ; print list commands
032601  63 67 51 64 43 50               SIXBIT  /SWITCH/                  ; print switch commands
032602  000000  000000                  0                                 ; end of list

032603  037 00 0 00 054546      TSTHLL: TMSGCD  <Tests Available:>

                                        TMSGD   <
                                ALL n   All tests (n times)
                                EBUS n  EBus module tests
                                SEQ n   SEQ related tests
                                ALU n   2901 related tests
                                MPROC n MPROC module tests
                                CBUS n  CBus module tests
                                MBUS n  MBUS related tests
032604  037 00 0 00 054553      XXm n   Test m, category XX (where XX can be EB,SE,AL,MP,CB)>

```

```
                                          TMSGD   <
                                  TAKE    Take cmds from file
                                  NTAKE   Take cmds from file (not echoed)
                                  DEBUG   Enter port debugger
                                  ENABLE n  Enable execution of test n
                                  DISABL n  Disable execution of test n
                                  DISLIS  List disabled tests
                                  DIAMON  Enter DIAMON
                                  DDT     Enter DDT
                                  HELP    Print this
032605  037 00 0 00 054637        EXIT    Exit program>

                                          TMSGD   <
                                  CONFIG  Determine/print configuration
                                  CONPNT  Print configuration
032606  037 00 0 00 054732        SELECT  Select device to test>

                                          TMSGD   <
                                  LIST n  List all tests (or just test n)
                                  LISTEB n  List EBus module tests (or test n)
                                  LISTSE n  List Sequencer related tests
                                  LISTAL n  List 2901's related tests
                                  LISTMP n  List MPROC module tests
032607  037 00 0 00 054757        LISTCB n  List CBus module tests>

                                          TMSGD   <
                                  Switches:  To complement switch, type switch name

                                  SWITCH  Enter switches
                                  SWPRIN  Print current switches

                                  NOPNT   LOOPER  RELIAB   TRACE   LOOPGM  DSPEAR  SDEBUG  NIPORT  MMPROC
                                  PNTLPT  ERSTOP  TXTINH   INHFLT  LOOPTS  UDEBUG  LDEBUG  CIPORT  MCBUS
032610  037 00 0 00 055037        DING    PALERS  OPRSEL   INHMSG  RUNALL  MDEBUG  IDEBUG>
032611  000000  000000                    0
```

```

                                        ;#**.*************************************************************
                                        ;* TSTDSA - Disable a test.
                                        ;* TSTENA - Enable a test.
                                        ;#*************************************************************

032612  402 00 0 00 064600'             TSTDSA: SETZM   ENATST#                 ; clear 'enable' flag
032613  200 00 0 00 055146                      MOVE    [FMSGCD (DISABL XXnnn XXnnn ... <CR>)]
032614  202 00 0 00 064645'                     MOVEM   TMP1#
032615  254 00 0 00 032621                      JRST    TSTENO                  ; continue
032616  476 00 0 00 064600'             TSTENA: SETOM   ENATST#                 ; set 'enable' flag
032617  200 00 0 00 055156                      MOVE    [FMSGCD (ENABLE XXnnn XXnnn ... <CR>)]
032620  202 00 0 00 064645'                     MOVEM   TMP1#
032621  336 00 0 00 032171*             TSTENO: SKIPN   ARGFLG                  ; argument given?
                                                JRST    [FMSGCD <? Missing argument>
                                                         GO      FIOFF
032622  254 00 0 00 055164                               RTN]

                                        ; Get an argument

032623  200 01 0 00 064645'             TSTDSO: MOVE    1,TMP1                  ; get help string
032624  260 17 0 00 032555*                     GO      .SARG                   ; yes - get SIXBIT argument
032625  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
032626  263 17 0 00 000000                       RTN                            ; no more arguments
032627  200 01 0 00 032560*                     MOVE    1,ARGUM                 ; get argument
032630  260 17 0 00 034315                      GO      TSTDEC                  ; decode as a test name
                                                 JRST   [FMSGCD <? Argument not recognizable as a test name 'XXn'>
                                                         GO      FIOFF
032631  254 00 0 00 054167                               RTN]
                                                 JRST   [FMSGC  <? Test selected is not in range 1->
                                                         MOVE    1              ; get range
                                                         PNTOCF                 ; print it
                                                         PCRLF
                                                         GO      FIOFF
032632  254 00 0 00 054202                               RTN]

                                        ; Now perform the specified function

032633  260 17 0 00 042775                      GO      TSTGET                  ; get test data
032634  200 01 0 00 047013                      MOVE    1,TSTPC                 ; get test PC
032635  200 02 0 00 047014                      MOVE    2,TSTFLG                ; get test flags
032636  332 00 0 00 064600'                     SKIPE   ENATST                  ; 'enable' flag set?
032637  621 02 0 00 000100                      TLZ     2,(TDENA)               ; yes - clear 'disable' bit
032640  336 00 0 00 064600'                     SKIPN   ENATST                  ; 'enable' flag set?
032641  661 02 0 00 000100                      TLO     2,(TDENA)               ; no - set 'disable' bit
032642  202 02 0 01 000001                      MOVEM   2,1(1)                  ; save flags + test address
032643  254 00 0 00 032623                      JRST    TSTDSO                  ; get next argument
```

```

                                        ;#*********************************************************************
                                        ;* TSTDSL - List disabled tests
                                        ;#*********************************************************************

032644  200 01 0 00 055173      TSTDSL: MOVE    1,[FMSGCD (DISLIS <CR>)]
032645  260 17 0 00 032304*             GO      CHKARG                  ; check for argument
032646  263 17 0 00 000000              RTN                             ; error - exit
032647  476 00 0 00 047023              SETOM   TSTINH                  ; initialize fault isolation flag
032650  201 00 0 00 033327              MOVEI   TDISP1-1                ; get address of dispatch table
032651  202 00 0 00 047015              MOVEM   TSTADD                  ; save it
032652  400 06 0 00 000000              SETZ    6,                      ; count of tests printed
032653  260 17 0 00 043023      TSTDLO: GO      SELTST                  ; select a test
                                         PJRST  [PCRL                   ; no more - return
032654  254 00 0 00 055174                       RTN]
032655  255 00 0 00 000000               JFCL                           ; not valid (ignore)
032656  200 00 0 00 047014              MOVE    TSTFLG                  ; get test flags
032657  607 00 0 00 000100              TLNN    (TDENA)                 ; disabled?
032660  254 00 0 00 032653              JRST    TSTDLO                  ; no - try next test
032661  377 00 0 00 000006              SOSG    6                       ; done with line yet?
                                        GO      [MOVEI  6,13            ; yes - reset counter and
                                                 PCRL                   ;   print a blank line
032662  260 17 0 00 055176                       RTN]
032663  200 01 0 00 047010              MOVE    1,TSTCLS                ; get test class
032664  200 00 0 01 031275              MOVE    INDLIS(1)               ; get class description
032665  037 00 0 00 000002              PNTSIX                          ; print it
032666  200 00 0 00 047011              MOVE    TSTNUM                  ; get test number
032667  037 16 0 00 000003              PNTOCS                          ; print it
032670  037 00 0 00 000040              PSP
032671  254 00 0 00 032653              JRST    TSTDLO                  ; continue
```

```
                                        ;#***************************************************************
                                        ;* TSTSWI/TSTSWP - Input switches / Print switch selections
                                        ;#***************************************************************

032672  200 01 0 00 055205      TSTSWI: MOVE    1,[FMSGCD (SWITCH <CR>)]
032673  260 17 0 00 032645*             GO      CHKARG                  ; check for argument
032674  263 17 0 00 000000              RTN                             ; error - exit
032675  260 17 0 00 000000*             GO      .ISWT                   ; input switches
032676  263 17 0 00 000000              RTN                             ; ask for a test

032677  200 01 0 00 055212      TSTSWP: MOVE    1,[FMSGCD (SWPRIN <CR>)]
032700  260 17 0 00 032673*             GO      CHKARG                  ; check for argument
032701  263 17 0 00 000000              RTN                             ; error - exit
032702  260 17 0 00 000000*             GO      .SWCHP                  ; print switches
032703  263 17 0 00 000000              RTN                             ; ask for a test


                                        ;#***************************************************************
                                        ;* TSTSSW - Switch complement commands
                                        ;#***************************************************************

032704  200 01 0 00 055220      TSTSSW: MOVE    1,[FMSGCD (Switch Name <CR>)]
032705  260 17 0 00 032700*             GO      CHKARG                  ; check for argument
032706  263 17 0 00 000000              RTN                             ; error - exit
032707  200 01 0 00 000000*             MOVE    1,.CGOT                 ; get dispatch number
032710  275 01 0 00 000036              SUBI    1,SWSTT                 ; normalize to 0..
032711  260 17 0 00 000000*             GO      SWCOM                   ; complement switch
032712  263 17 0 00 000000              RTN                             ; continue
```

```
                                ;#*****************************************************************
                                ;* TSTTAK - Take commands from a file
                                ;* TSTTAN - Take commands from a file without echoing results
                                ;
                                ; This works by reading commands from a file as if they were being
                                ; typed to the diagnostic directly.  Either end of file or the first
                                ; occurrence of an error halts this procedure.
                                ;#*****************************************************************

032713  402 00 0 00 064602'     TSTTAN: SETZM   FINECH                  ; clear echo switch
032714  334 00 0 00 000000              SKIPA                           ; continue
032715  476 00 0 00 064602'     TSTTAK: SETOM   FINECH                  ; set echo switch
032716  200 01 0 00 055231              MOVE    1,[FMSGCD (TAKE <CR> or TAKE Fname Ext <CR>)]
032717  260 17 0 00 032624*             GO      .SARG                   ; get SIXBIT argument
032720  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
032721  254 00 0 00 032732               JRST   TSTTA0                  ; no more arguments
032722  200 01 0 00 032627*             MOVE    1,ARGUM                 ; get argument
032723  202 01 0 00 032747              MOVEM   1,TAKFIL                ; save file name
032724  210 01 0 00 055231              MOVN    1,[FMSGCD (TAKE <CR> or TAKE Fname Ext <CR>)]
032725  260 17 0 00 032717*             GO      .SARG                   ; get SIXBIT argument
032726  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
032727  254 00 0 00 032732               JRST   TSTTA0                  ; no more arguments
032730  200 01 0 00 032722*             MOVE    1,ARGUM                 ; get argument
032731  202 01 0 00 032750              MOVEM   1,TAKFIL+1              ; save file name extension

                                ; Now start reading the file specified

032732  260 17 0 00 031065*     TSTTA0: GO      FIOFF                   ; shut off previous file input
032733  201 00 0 00 032747              MOVEI   TAKFIL                  ; point to file name buffer
032734  037 05 0 00 000004              FSELECT                         ; init the file
                                        JRST    [FMSGC  <? Can't access ">
                                                 MOVE   TAKFIL                  ; get file name
                                                 PNTSXF                         ; print it
                                                 PNTCIF "."
                                                 MOVE   TAKFIL+1                ; get extension
                                                 PNTSXF                         ; print it
                                                 FMSGD  <" on selected load device.>
032735  254 00 0 00 055244                       RTN]
032736  037 01 0 00 055254              FMSGC   <[File input from: >
032737  200 00 0 00 032747              MOVE    TAKFIL                  ; get the selected file name
032740  037 01 0 00 000002              PNTSXF                          ; print it
032741  037 01 0 00 000056              PNTCIF  "."
032742  200 00 0 00 032750              MOVE    TAKFIL+1                ; get the extension of file
032743  037 01 0 00 000002              PNTSXF                          ; print it
032744  037 01 0 00 054523              FMSGD   <]>
032745  476 00 0 00 064603'             SETOM   FINPUT                  ; set 'file input' flag
032746  263 17 0 00 000000              RTN                             ; exit

                                ; File name

032747  44 46 60 64 41 00       TAKFIL: SIXBIT  /DFPTA/
032750  43 55 44 00 00 00               SIXBIT  /CMD/

```

```
                                                SUBTTL  Test Dispatching

                                        ;#**********************************************************************
                                        ;* TSTSPC - Run only one test
                                        ;#**********************************************************************

032751  201 00 0 00 033265              TSTSPC: MOVEI   TSTALT                  ; set up altmode transfer address
032752  202 00 0 00 030063                      MOVEM   ALTMGO                  ;   to exit port testing
032753  402 00 0 00 047020                      SETZM   TSTREP                  ; clear pass count
032754  260 17 0 00 032327*                     GO      SWITT                   ; get program switches
032755  602 00 0 00 020000                      TRNE    LOOPTS                  ; loop on test?
                                                JRST    [HRLEI  377777          ; yes - get 'infinite' repeat count
032756  254 00 0 00 055261                               JRST   .+2]
032757  201 00 0 00 000001                      MOVEI   1                       ; no - set repeat count requested
032760  202 00 0 00 047021                      MOVEM   TSTREQ                  ;   to 1 pass
032761  402 00 0 00 047022                      SETZM   TSTMUL                  ; clear multiple test flag
032762  476 00 0 00 047023                      SETOM   TSTINH                  ; initialize fault isolation flag
032763  336 00 0 00 032621*                     SKIPN   ARGFLG                  ; argument given?
032764  254 00 0 00 032773                      JRST    TSTSP0                  ; no - do the test
032765  210 01 0 00 055271                      MOVN    1,[FMSGCD (XXnnn RepeatCount <CR>)]
032766  260 17 0 00 032240*                     GO      .DARG                   ; yes - get repeat count argument
032767  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
032770  334 00 0 00 000000                        SKIPA                         ; no argument given
032771  200 00 0 00 032730*                     MOVE    ARGUM                   ; get argument
032772  202 00 0 00 047021                      MOVEM   TSTREQ                  ; save it
032773  476 00 0 00 032763*             TSTSP0: SETOM   ARGFLG                  ; remember that we had an argument
032774  254 00 0 00 033122                      JRST    TSTD3                   ; do the test


                                        ;#**********************************************************************
                                        ;* RANGEN - Random number generator
                                        ;#**********************************************************************

032775  261 17 0 00 000001              RANGEN: RPUT    (1,2)                   ; save AC's
032777  200 01 0 00 064631'                     MOVE    1,RANNUM                ; get base number
033000  270 01 0 00 030022                      ADD     1,RANDBS                ; modify number in AC1
033001  241 01 0 00 777774                      ROT     1,-4                    ; to create a new number
033002  447 01 0 00 064631'                     EQVB    1,RANNUM                ; put result in AC1 and RANNUM
033003  405 01 0 00 007777                      ANDI    1,7777                  ; save only 4 digits
033004  220 01 0 17 777777                      IMUL    1,-1(P)                 ; obtain (0..n-1)*10000
033005  242 01 0 00 777764                      LSH     1,-^D12                 ; obtain 0..n-1
033006  350 00 0 00 000001                      AOS     1                       ; obtain 1..n
033007  313 01 0 17 777777                      CAMLE   1,-1(P)                 ; in range?
033010  200 01 0 17 777777                      MOVE    1,-1(P)                 ; no - force it to be
033011  250 01 0 17 777777                      EXCH    1,-1(P)                 ; set up returned argument
033012  262 17 0 00 000002                      RGET    (2,1)                   ; restore AC's
033014  263 17 0 00 000000                      RTN
```

```

                                          ;#*********************************************************************
                                          ; TSTRAN - Run tests randomly
                                          ;#*********************************************************************

033015  201 00 0 00 033040                TSTRAN: MOVEI   TSTRAA                  ; set up altmode transfer address
033016  202 00 0 00 030063                        MOVEM   ALTMGO                  ;   to exit port testing
033017  402 00 0 00 047020                        SETZM   TSTREP                  ; clear pass count
033020  201 00 0 00 000001                        MOVEI   1                       ; set repeat count requested
033021  202 00 0 00 047021                        MOVEM   TSTREQ                  ;   to 1 pass
033022  402 00 0 00 047022                        SETZM   TSTMUL                  ; clear multiple test flag
033023  476 00 0 00 047023                        SETOM   TSTINH                  ; initialize fault isolation flag
033024  402 00 0 00 064630'                       SETZM   RANCNT#                 ; initialize test count
033025  400 03 0 00 000000                        SETZ    3,                      ; set a a base number using the
033026  260 17 0 00 000000*                       GO      RUNTME                  ;   program run time and save
033027  200 00 0 00 000000*                       MOVE    CURTIM                  ;   it in RANNUM
033030  202 00 0 00 064631'                       MOVEM   RANNUM

                                          ; Determine test to run

033031  201 01 0 00 000351                TSTRA0: MOVEI   1,TEND-TDISP1           ; get range 1..MAXTST
033032  260 17 0 00 032775                        GO      RANGEN                  ; get a random number (in AC1)
033033  201 01 0 01 033326                        MOVEI   1,TDISP1-2(1)           ; get test dispatch address
033034  202 01 0 00 047015                        MOVEM   1,TSTADD                ; save it (test - 1)
033035  260 17 0 00 033122                        GO      TSTD3                   ; do the test
033036  350 01 0 00 064630'                       AOS     1,RANCNT                ; increment test count
033037  254 00 0 00 033031                        JRST    TSTRA0                  ; keep looping

                                          ; Altmode exit

033040  037 01 0 00 055272                TSTRAA: FMSGC   <[Testing aborted after >
033041  200 00 0 00 064630'                       MOVE    RANCNT                  ; get test count
033042  037 15 0 00 000001                        PNTDCF                          ; print it
033043  037 01 0 00 055300                        FMSGD   < tests]>
033044  262 17 0 00 064660'                       GET     XXX#                    ; pop off extra item
033045  263 17 0 00 000000                        RTN                             ; exit
```

```

                                     ;#**********************************************************************
                                     ;* TSTDIS - Dispatch to the tests
                                     ;#**********************************************************************

                                     ; Set up flags, defaults, etc.

 033046  201 00 0 00 033265          TSTDIS: MOVEI   TSTALT                  ; set up altmode transfer address
 033047  202 00 0 00 030063                  MOVEM   ALTMGO                  ;   to exit port testing
 033050  402 00 0 00 047020                  SETZM   TSTREP                  ; clear pass count
 033051  402 00 0 00 047021                  SETZM   TSTREQ                  ; set repeat count requested to 1
 033052  350 00 0 00 047021                  AOS     TSTREQ                  ;   pass
 033053  476 00 0 00 047022                  SETOM   TSTMUL                  ; set multiple test flag
 033054  336 00 0 00 032773*                 SKIPN   ARGFLG                  ; argument given?
 033055  254 00 0 00 033065                  JRST    TSTD0                   ; no - do the tests
 033056  210 01 0 00 052635                  MOVN    1,[FMSGCD (Cmd RepeatCount <CR>)]
 033057  260 17 0 00 032766*                 GO      .DARG                   ; yes - get repeat count argument
 033060  263 17 0 00 000000                   RTN                            ; error/altmode/question - exit
 033061  334 00 0 00 000000                    SKIPA                         ; no argument given
 033062  200 00 0 00 032771*                 MOVE    ARGUM                   ; get argument
 033063  202 00 0 00 047021                  MOVEM   TSTREQ                  ; save it
 033064  476 00 0 00 033054*                 SETOM   ARGFLG                  ; remember that we had an argument

                                     ; If OPRSEL switch is not set then all tests are run.

 033065  260 17 0 00 032754*          TSTD0:  GO      SWITT                   ; get the pgm switches
 033066  135 01 0 00 055302                  LDB     1,[POINT 3,0,35]        ; get only MMPROC and MCBUS switches
                                             XCT     [JFCL                   ; print module status
                                                       TMSGCD <[Assuming CBUS module is missing]>
                                                        TMSGCD <[Assuming MPROC and CBUS modules are missing]>
 033067  256 00 0 01 055325                             TMSGCD <[Assuming MPROC and CBUS modules are missing]>](1)
 033070  402 00 0 00 047023                  SETZM   TSTINH                  ; initialize fault isolation flag
 033071  602 00 0 00 200000                  TRNE    INHFLT                  ; inhibiting fault isolation?
 033072  476 00 0 00 047023                  SETOM   TSTINH                  ; yes - set the flag
 033073  201 00 0 00 033327                  MOVEI   TDISP1-1                ; get address without fault isolation
 033074  336 00 0 00 047023                  SKIPN   TSTINH                  ; inhibit fault isolation flag set?
 033075  201 00 0 00 033701                  MOVEI   TDISP2-1                ; no - get address with fault isolation
 033076  202 00 0 00 047015                  MOVEM   TSTADD                  ; set up test dispatch table address
 033077  607 00 0 00 000010                  TLNN    OPRSEL                  ; OPRSEL set?
 033100  254 00 0 00 033122                  JRST    TSTD3                   ; no - go run all of them

                                     ; Check for MBUS testing (if so, the pass count has already been obtained)

 033101  200 00 0 00 047017                  MOVE    TSTMSK                  ; get test mask
 033102  316 00 0 00 053072                  CAMN    [MBUS]                  ; MBUS command given?
 033103  336 00 0 00 064612'                 SKIPN   MBARGF                  ; 'MBUS argument given' flag set?
 033104  254 00 0 00 033107                  JRST    TSTD1                   ; no - continue
 033105  200 00 0 00 064611'                 MOVE    MBARG                   ; get repeat count
 033106  254 00 0 00 033121                  JRST    TSTD2                   ; continue
```

```
                                             ; Handle pass count

033107  200 00 0 00 047021           TSTD1:  MOVE    TSTREQ                  ; get argument typed
033110  332 00 0 00 033064*                  SKIPE   ARGFLG                  ; any argument given?
033111  254 00 0 00 033121                   JRST    TSTD2                   ; yes - skip over this question
033112  332 00 0 00 064603'                  SKIPE   FINPUT                  ; file input switch set?
                                             JRST    [MOVEI  1               ; yes - default to 1 iteration
033113  254 00 0 00 055331                            JRST   TSTD2]
033114  037 01 0 00 055333                   FMSGC   <How many passes? (CR to loop forever) - >
033115  037 04 0 00 000003                   TTIDEC
033116  007 00 0 00 033107                   ALTCHK  TSTD1
033117  336 00 0 00 030510                   SKIPN   TTNBRF                  ; CR typed?
033120  200 00 0 00 055344                   MOVE    [377777,,777777]        ; yes - loop forever
033121  202 00 0 00 047021           TSTD2:  MOVEM   TSTREQ                  ; save count

                                             ; Save test class/number selection for multiple pass use

033122  200 00 0 00 047015           TSTD3:  MOVE    TSTADD                  ; get test dispatch address
033123  202 00 0 00 064634'                  MOVEM   SAVADD#                 ; save test dispatch address
033124  476 00 0 00 047034                   SETOM   PMODE                   ; set program mode to 'test'

                                             ; Initialize test class/number selection

033125  200 00 0 00 064634'          TSTD4:  MOVE    SAVADD                  ; get test dispatch address
033126  202 00 0 00 047015                   MOVEM   TSTADD                  ; save test dispatch address

                                             ; Initialize device number

033127  476 00 0 00 047005                   SETOM   PORSEL                  ; initialize select word
```

```

                                          ; Select next device

033130  400 00 0 00 000000        TSTD5:  SETZ                            ; clear select word
033131  200 01 0 00 047005                MOVE    1,PORSEL                ; get current port
033132  306 01 0 00 000000                CAIN    1,0                     ; NI port?
033133  200 00 0 00 047003                MOVE    NISEL                   ; get NI select word
033134  306 01 0 00 000001                CAIN    1,1                     ; CI port?
033135  200 00 0 00 047004                MOVE    CISEL                   ; yes - get CI select word
033136  602 00 0 00 000004                TRNE    4                       ; selected?
033137  260 17 0 00 033304                GO      TSTEOP                  ; yes - print end of pass message
033140  602 00 0 00 000004                TRNE    4                       ; selected?
033141  260 17 0 00 031515                GO      ISOPNT                  ; yes - print fault isolation data
033142  350 01 0 00 047005                AOS     1,PORSEL                ; point to next device
033143  303 01 0 00 000001                CAILE   1,1                     ; done yet?
033144  254 00 0 00 033153                JRST    TSTD6                   ; yes - do next pass
033145  200 00 0 00 047003                MOVE    NISEL                   ; get NI select word
033146  306 01 0 00 000001                CAIN    1,1                     ; CI device?
033147  200 00 0 00 047004                MOVE    CISEL                   ; yes - get CI select word
033150  606 00 0 00 000004                TRNN    4                       ; selected?
033151  254 00 0 00 033130                JRST    TSTD5                   ; no - check next device
033152  254 00 0 00 033165                JRST    TSTD7                   ; yes - go test

                                          ; Decrement pass count

033153  260 17 0 00 031515        TSTD6:  GO      ISOPNT                  ; print fault isolation data
033154  350 01 0 00 047020                AOS     1,TSTREP                ; increment pass count
033155  260 17 0 00 033065*               GO      SWITT                   ; get switches
033156  336 00 0 00 030037                SKIPN   USER                    ; user mode?
033157  607 00 0 00 400000                TLNN    ABORT                   ; abort switch set?
033160  334 00 0 00 000000                SKIPA                           ; no - continue (or user mode)
033161  263 17 0 00 000000                RTN                             ; exit
033162  373 00 0 00 047021                SOSLE   TSTREQ                  ; passes all done?
033163  254 00 0 00 033125                JRST    TSTD4                   ; no - keep looping
033164  263 17 0 00 000000                RTN                             ; exit

                                          ; Initialize fault isolation data area

033165  200 16 0 00 047001        TSTD7:  MOVE    MBCN,PORTNI             ; get NI device code
033166  332 00 0 00 047005                SKIPE   PORSEL                  ; CI port selected?
033167  200 16 0 00 047002                MOVE    MBCN,PORTCI             ; yes - get CI device code
033170  202 16 0 00 047006                MOVEM   MBCN,UUT                ; save unit under test
033171  260 17 0 00 032354*               GO      DEVREQ                  ; request the device
033172  254 00 0 00 033130                JRST    TSTD5                   ; failed - exit pass
033173  402 00 0 00 031753                SETZM   NETBAD                  ; clear first word
033174  200 00 0 00 055345                MOVE    [NETBAD,,NETBAD+1]      ; build BLT pointer
033175  251 00 0 00 032062                BLT     NETBAD+^D71             ; clear the areas
033176  402 00 0 00 064601'               SETZM   ERRCNT#                 ; clear error count
033177  402 00 0 00 064604'               SETZM   FLTPNT                  ; clear 'isolation data printed' flag
```

```
                                        ; Initialze test class/number at beginning of pass

033200  200 00 0 00 064634'                     MOVE    SAVADD                          ; get test dispatch address
033201  202 00 0 00 047015                      MOVEM   TSTADD                          ; save it
033202  260 17 0 00 033155*                     GO      SWITT                           ; get the pgm switches
033203  201 01 0 00 000001                      MOVEI   1,1                             ; initialize repeat count
033204  603 00 0 00 000400                      TLNE    RELIAB                          ; reliability switch set?
033205  201 01 0 00 000004                      MOVEI   1,4                             ; yes - set up count of 4
033206  202 01 0 00 047027                      MOVEM   1,TSTREL                        ; save it

                                        ; Now actually do the tests

033207  200 00 0 00 047027              TSTD8:  MOVE    TSTREL                          ; get reliability count
033210  202 00 0 00 047030                      MOVEM   TSTCNT                          ; save it
033211  260 17 0 00 043023                      GO      SELTST                          ; select next test
033212  254 00 0 00 033130                       JRST   TSTD5                           ; end of tests
033213  254 00 0 00 033255                       JRST   TSTD12                          ; not a valid test
033214  200 16 0 00 047006              TSTD9:  MOVE    MBCN,UUT                        ; valid test - set up MBCN
033215  260 17 1 00 047013                      GO      @TSTPC                          ; dispatch to test
033216  254 00 0 00 033255                      JRST    TSTD12                          ; disabled test - skip it
033217  332 00 0 00 047023                      SKIPE   TSTINH                          ; inhibit fault isolation?
033220  254 00 0 00 033246                      JRST    TSTD11                          ; yes - continue
033221  332 00 0 00 064606'                     SKIPE   IDEBUG                          ; fault isolation debug mode?
033222  260 17 0 00 034267                      GO      FLTCHK                          ; yes - check if this test should fail
033223  322 15 0 00 033246                      JUMPE   ERFLG,TSTD11                    ; test did not fail - continue
```

```

                                        ; If this is the first failure, also set up new test dispatch table (as
                                        ; specified in the test itself).

033224  332 00 0 00 064601'                     SKIPE   ERRCNT                  ; any errors yet?
033225  254 00 0 00 033231                      JRST    TSTD10                  ; no - continue
033226  200 00 0 00 047013                      MOVE    TSTPC                   ; get test PC
033227  271 00 0 00 000003                      ADDI    3                       ; point to new table address
033230  202 00 0 00 047015                      MOVEM   TSTADD                  ; set up new dispatch address

                                        ; Handle fault isolation

033231  350 00 0 00 064601'             TSTD10: AOS     ERRCNT                  ; increment error count
033232  200 03 0 00 047024                      MOVE    3,TSTHIG                ; get addr of high prob module list
033233  201 04 0 00 000014                      MOVEI   4,^D12                  ; weight of a high prob module
033234  332 00 0 00 064606'                     SKIPE   IDEBUG                  ; fault isolation debug mode?
                                                GO      [TMSGC  <*** >          ; yes - print header ...
                                                         MOVE   1,TSTCLS        ; get test class
                                                         MOVE   INDLIS(1)       ; get class description
                                                         PNTSIX                 ; print it
                                                         MOVE   TSTNUM          ; get test number
                                                         PNTOCS                 ; print it
                                                         TMSG   < ***>
                                                         TMSGC  <High networks:  >
                                                         RTN]
033235  260 17 0 00 055355
033236  260 17 0 00 034251                      GO      FLTHAN                  ; go to fault handler
033237  200 03 0 00 047025                      MOVE    3,TSTLOW                ; get addr of low prob module list
033240  201 04 0 00 000006                      MOVEI   4,^D6                   ; weight of a low prob module
033241  332 00 0 00 064606'                     SKIPE   IDEBUG                  ; fault isolation debug mode?
033242  037 00 0 00 055366                      TMSGC   <Low networks:  >       ; yes - print header
033243  260 17 0 00 034251                      GO      FLTHAN                  ; go to fault handler
033244  332 00 0 00 064606'                     SKIPE   IDEBUG                  ; fault isolation debug mode?
033245  037 00 0 00 030242                      PCRL                            ; yes - end with a CRLF
```

```
                                  ; Dispatch is done - now check altmode

033246  332 00 0 00 064570'       TSTD11: SKIPE   ALTF                    ; 'altmode typed' flag set?
033247  254 00 1 00 030063                JRST    @ALTMGO                 ; yes - exit
033250  037 07 0 00 000003                TTALTM                          ; altmode typed?
033251  334 00 0 00 000000                SKIPA                           ; no - continue
033252  254 00 1 00 030063                JRST    @ALTMGO                 ; yes - exit thru $ transfer address

                                  ; Loop till reliability count expired

033253  373 00 0 00 047030                SOSLE   TSTCNT                  ; decrement reliability count
033254  254 00 0 00 033214                JRST    TSTD9                   ; loop till done
033255  332 00 0 00 030037        TSTD12: SKIPE   USER                    ; user mode?
033256  254 00 0 00 033262                JRST    .+4                     ; yes - continue
033257  260 17 0 00 033202*               GO      SWITT                   ; get switches
033260  603 00 0 00 200000                TLNE    RSTART                  ; RSTART switch set?
033261  254 00 0 00 030610                JRST    START                   ; yes - restart the diagnostic
033262  336 00 0 00 047022                SKIPN   TSTMUL                  ; multiple tests being done?
033263  254 00 0 00 033130                JRST    TSTD5                   ; no - check next device
033264  254 00 0 00 033207                JRST    TSTD8                   ; yes - do next test

                                  ; Altmode transfer code

033265  037 01 0 00 055372        TSTALT: FMSGC   <[Testing aborted>
033266  336 00 0 00 047020                SKIPN   TSTREP                  ; zero passes?
033267  254 00 0 00 033274                JRST    TSTALO                  ; yes - don't print how many
033270  037 01 0 00 055376                FMSG    < after >
033271  200 00 0 00 047020                MOVE    TSTREP                  ; get pass count
033272  037 15 0 00 000001                PNTDCF                          ; print it
033273  037 01 0 00 055400                FMSG    < passes>
033274  037 01 0 00 054523        TSTALO: FMSGD   <]>
033275  260 17 0 00 032732*               GO       FIOFF                  ; clear 'file input' switch
033276  476 00 0 00 064570'               SETOM   ALTF                    ; set altmode typed flag
033277  260 17 0 00 031515                GO      ISOPNT                  ; yes - print final flt isolation data
033300  260 17 0 00 033257*               GO      SWITT                   ; get program switches
033301  607 00 0 00 000010                TLNN    OPRSEL                  ; OPRSEL switch set?
033302  254 00 0 00 032265                JRST    TSTDIA                  ; no - exit to DIAMON
033303  263 17 0 00 000000                RTN                             ; exit
```

```

                                      ; End of pass message

033304  336 00 0 00 047022            TSTEOP: SKIPN   TSTMUL                  ; multiple tests being run?
033305  263 17 0 00 000000                    RTN                             ; no - return
033306  261 17 0 00 000000                    RPUT    (0,1)                   ; save AC0 and AC1
033310  260 17 0 00 031576*                   GO      PNTDEV
033311  037 00 0 00 055402                    TMSG    < End Pass >            ; yes - print pass count
033312  260 17 0 00 033300*                   GO      SWITT                   ; get switches
033313  607 00 0 00 000010                    TLNN    OPRSEL                  ; operator select?
033314  334 01 0 00 030047                    SKIPA   1,PASCNT                ; Nope, Skip and load SUBRTN pass cnt
033315  200 01 0 00 047020                    MOVE    1,TSTREP                ; Yep get local pass count
033316  201 00 0 01 000001                    MOVEI   0,1(1)                  ; Get the "real" pass count
033317  037 15 0 00 000000                    PNTDEC
033320  037 00 0 00 055405                    TMSG    <. at >
033321  402 00 0 00 030640*                   SETZM   FORPNT                  ; clear force print
033322  205 03 0 00 255000                    MOVSI   3,(JFCL)                ; set up timer routine
033323  260 17 0 00 033026*                   GO      RUNTME                  ; print current time
033324  037 00 0 00 030242                    PCRL
033325  262 17 0 00 000001                    RGET    (1,0)                   ; restore AC0
033327  263 17 0 00 000000                    RTN
```

```

                                        ;#**********************************************************************
                                        ;* Test Dispatch List - in sequential order
                                        ;#**********************************************************************

                        000000          ZEB==0B2                                ; EBUS test
                100000  000000          ZSE==1B2                                ; SEQ test
                200000  000000          ZAL==2B2                                ; ALU test
                300000  000000          ZMP==3B2                                ; MPROC test
                400000  000000          ZCB==4B2                                ; CBUS test

                                        ; EBUS Tests

        033330                          TEBUS:
        033330  000000000000#           TDISP1: TSTE1           ; EB1 - CONI Works?
        033331  000000000000#                   TSTE2           ; EB2 - RH20/IPA Interaction
        033332  000000000000#                   TSTE3           ; EB3 - RH20/IPA Interaction
        033333  000000000000#                   TSTE4           ; EB4 - RH20/IPA Interaction
        033334  000000000000#                   TSTE5           ; EB5 - RH20/IPA Interaction
        033335  000000000000#                   TSTE6           ; EB6 - CSR Cleared After Reset
        033336  000000000000#                   TSTE7           ; EB7 - Read/Write CSR Bits
        033337  000000000000#                   TSTE10          ; EB10 - CSR Bits Interaction
        033340  000000000000#                   TSTE11          ; EB11 - EBUF Access Test
        033341  000000000000#                   TSTE12          ; EB12 - EBUF/CSR Conflict Test
        033342  000000000000#                   TSTE13          ; EB13 - KMUX Parity Generators
        033343  000000000000#                   TSTE14          ; EB14 - EBUF/CSR Conflict Test
        033344  000000000000#                   TSTE15          ; EB15 - EBUF Reset Test
        033345  000000000000#                   TSTE16          ; EB16 - EBUF Data Path Test
        033346  000000000000#                   TSTE17          ; EB17 - CSR Bit 18 Self Clearing Test
        033347  000000000000#                   TSTE20          ; EB20 - CSR Bit 18 Reset Test
        033350  000000000000#                   TSTE21          ; EB21 - CSR Bit 18 Reset Test
        033351  000000000000#                   TSTE22          ; EB22 - Read/Write EBUF When Ucode Running
        033352  000000000000#                   TSTE23          ; EB23 - Par Gen - Not Forcing an Error
        033353  000000000000#                   TSTE24          ; EB24 - Par Gen - Forcing an Error
        033354  000000000000#                   TSTE25          ; EB25 - Read only CSR Bits
        033355  000000000000#                   TSTE26          ; EB26 - CSR 01 'DIAG RQST CSR'
        033356  000000000000#                   TSTE27          ; EB27 - CSR 01 'DIAG RQST CSR'
        033357  000000000000#                   TSTE30          ; EB30 - CSR 01 'DIAG RQST CSR'
        033360  000000000000#                   TSTE31          ; EB31 - CSR 01 'DIAG RQST CSR'
        033361  000000000000#                   TSTE32          ; EB32 - CSR 01 'DIAG RQST CSR'
        033362  000000000000#                   TSTE33          ; EB33 - CSR 01 'DIAG RQST CSR'
        033363  000000000000#                   TSTE34          ; EB34 - CSR 02 'DIAG CSR CHNG'
        033364  000000000000#                   TSTE35          ; EB35 - Constant MUX
        033365  000000000000#                   TSTE36          ; EB36 - Constant MUX Interference
        033366  000000000000#                   TSTE37          ; EB37 - CSR Bit 24 - EBUS PE Occurs at Wrong Time
        033367  000000000000#                   TSTE40          ; EB40 - EBUS Transfer via DATAO's
        033370  000000000000#                   TSTE41          ; EB41 - EBUS Transfer via DATAI's
        033371  000000000000#                   TSTE42          ; EB42 - CSR 05 'Rqst Interrupt'
        033372  000000000000#                   TSTE43          ; EB43 - CSR 05 'Rqst Interrupt'
        033373  000000000000#                   TSTE44          ; EB44 - CSR 05 'Rqst Interrupt'
        033374  000000000000#                   TSTE45          ; EB45 - CSR 06 'CRAM Par Err'
        033375  000000000000#                   TSTE46          ; EB46 - CSR 07 'MBUS Err'
        033376  000000000000#                   TSTE47          ; EB47 - CSR 11 'Spare' Bit
        033377  000000000000#                   TSTE50          ; EB50 - CSR 12 'Disable Complete' Bit
```

```
033400  000000000000#                      TSTE51          ; EB51 - CSR 13 'Enable Complete' Bit
033401  000000000000#                      TSTE52          ; x
033402  000000000000#                      TSTE53          ; EB53 - MPROC Run Bit
033403  000000000000#                      TSTE54          ; EB54 - Basic KMUX Parity
033404  000000000000#                      TSTE55          ; EB55 - Clearing MPROC Run Bit
033405  000000000000#                      TSTE56          ; EB56 - Clearing MPROC Run Bit
033406  000000000000#                      TSTE57          ; EB57 - Verify 'Diag Single Cyc'
033407  000000000000#                      TSTE60          ; EB60 - PI Level n/0 Interrupts
033410  000000000000#                      TSTE61          ; EB61 - Check PIA Level 0
033411  000000000000#                      TSTE62          ; EB62 - Check Interrupt on Level 1-7
033412  000000000000#                      TSTE63          ; EB63 - CSR 05 'Rqst Interrupt' Clearing
033413  000000000000#                      TSTE64          ; EB64 - IOP Function 0 - 40+2N Interrupt
033414  000000000000#                      TSTE65          ; EB65 - IOP Function 1 - 40+2N Interrupt
033415  000000000000#                      TSTE66          ; x
033416  000000000000#                      TSTE67          ; x
033417  000000000000#                      TSTE70          ; EB70 - IOP Function 4 - Examine
033420  000000000000#                      TSTE71          ; EB71 - IOP Function 5 - Deposit
033421  000000000000#                      TSTE72          ; EB72 - IOP Function 7 - Examine/Increment
033422  000000000000#                      TSTE73          ; EB73 - CSR 25-26,28 KL side
033423  000000000000#                      TSTE74          ; EB74 - CSR 25-26,28 Port side
033424  000000000000#                      TSTE75          ; EB75 - CSR 27,30-31 KL side
033425  000000000000#                      TSTE76          ; EB76 - CSR 27,30-31 Port side

                                   ; SEQ Tests

033426  000000000000#              TSEQ:   TSTS1           ; SE1 - JMAP Instruction
033427  000000000000#                      TSTS2           ; SE2 - JZ Instruction
033430  000000000000#                      TSTS3           ; SE3 - JZ Instruction - Effect on Reg/Ctr
033431  000000000000#                      TSTS4           ; SE4 - JZ Instruction - Effect on Reg/Ctr
033432  000000000000#                      TSTS5           ; SE5 - CONT Instruction
033433  000000000000#                      TSTS6           ; SE6 - CJP Instruction
033434  000000000000#                      TSTS7           ; SE7 - CJV Instruction
033435  000000000000#                      TSTS10          ; SE10 - CJS Instruction
033436  000000000000#                      TSTS11          ; SE11 - CJS Instruction
033437  000000000000#                      TSTS12          ; SE12 - CJS Instruction
033440  000000000000#                      TSTS13          ; SE13 - CRTN Instruction
033441  000000000000#                      TSTS14          ; SE14 - CRTN Instruction
033442  000000000000#                      TSTS15          ; SE15 - Stack location #1
033443  000000000000#                      TSTS16          ; SE16 - Stack location #2
033444  000000000000#                      TSTS17          ; SE17 - Stack location #3
033445  000000000000#                      TSTS20          ; SE20 - Stack location #4
033446  000000000000#                      TSTS21          ; SE21 - Stack location #5
033447  000000000000#                      TSTS22          ; SE22 - Increment/Decrement Stack Pointer
033450  000000000000#                      TSTS23          ; SE23 - Increment/Decrement Stack Pointer
033451  000000000000#                      TSTS24          ; SE24 - Increment/Decrement Stack Pointer
033452  000000000000#                      TSTS25          ; SE25 - Increment/Decrement Stack Pointer
033453  000000000000#                      TSTS26          ; SE26 - Increment/Decrement Stack Pointer
033454  000000000000#                      TSTS27          ; SE27 - Overflowing the Stack
033455  000000000000#                      TSTS30          ; SE30 - CJPP Instruction
```

```
2028  033456  000000000000#              TSTS31          ; SE31 - CJPP Instruction
2029  033457  000000000000#              TSTS32          ; SE32 - CJPP Instruction
2030  033460  000000000000#              TSTS33          ; SE33 - LDCT Instruction
2031  033461  000000000000#              TSTS34          ; SE34 - LDCT Instruction
2032  033462  000000000000#              TSTS35          ; SE35 - Register/Counter
2033  033463  000000000000#              TSTS36          ; SE36 - JRP Instruction
2034  033464  000000000000#              TSTS37          ; SE37 - JSRP Instruction
2035  033465  000000000000#              TSTS40          ; SE40 - JSRP Instruction
2036  033466  000000000000#              TSTS41          ; SE41 - JSRP Instruction
2037  033467  000000000000#              TSTS42          ; SE42 - PUSH Instruction
2038  033470  000000000000#              TSTS43          ; SE43 - PUSH Instruction
2039  033471  000000000000#              TSTS44          ; SE44 - PUSH Instruction
2040  033472  000000000000#              TSTS45          ; SE45 - LOOP Instruction
2041  033473  000000000000#              TSTS46          ; SE46 - LOOP Instruction
2042  033474  000000000000#              TSTS47          ; SE47 - LOOP Instruction
2043  033475  000000000000#              TSTS50          ; SE50 - RFCT Instruction
2044  033476  000000000000#              TSTS51          ; SE51 - RFCT Instruction
2045  033477  000000000000#              TSTS52          ; SE52 - RFCT Instruction
2046  033500  000000000000#              TSTS53          ; SE53 - RFCT Instruction
2047  033501  000000000000#              TSTS54          ; SE54 - RPCT Instruction
2048  033502  000000000000#              TSTS55          ; SE55 - RPCT Instruction
2049  033503  000000000000#              TSTS56          ; SE56 - TWB Instruction
2050  033504  000000000000#              TSTS57          ; SE57 - TWB Instruction
2051  033505  000000000000#              TSTS60          ; SE60 - TWB Instruction
2052  033506  000000000000#              TSTS61          ; SE61 - TWB Instruction
2053  033507  000000000000#              TSTS62          ; SE62 - TWB Instruction
2054  033510  000000000000#              TSTS63          ; SE63 - TWB Instruction
2055  033511  000000000000#              TSTS64          ; SE64 - Stack Interference Tests
2056  033512  000000000000#              TSTS65          ; SE65 - Stack Interference Tests
2057  033513  000000000000#              TSTS66          ; SE66 - Reg/Ctr Interference Tests
2058  033514  000000000000#              TSTS67          ; SE67 - Reg/Ctr Interference Tests
2059  033515  000000000000#              TSTS70          ; SE70 - Full Speed Sequencer Test
2060
2061                              ; ALU Tests
2062
2063  033516  000000000000#       TALU:  TSTA1           ; AL1 - 2901 0's Test
2064  033517  000000000000#              TSTA2           ; AL2 - 2901 1's Test
2065  033520  000000000000#              TSTA3           ; AL3 - 2901 OR Test - D OR 0
2066  033521  000000000000#              TSTA4           ; AL4 - 2901 AND Test - D AND 0
2067  033522  000000000000#              TSTA5           ; AL5 - 2901 NOT_R_AND_S Test - D NOT_RS 0
2068  033523  000000000000#              TSTA6           ; AL6 - 2901 XOR Test - D XOR 0
2069  033524  000000000000#              TSTA7           ; AL7 - 2901 XNOR Test - D XNOR 0
2070  033525  000000000000#              TSTA10          ; AL10 - 2901 R+S Test - D + 0
2071  033526  000000000000#              TSTA11          ; AL11 - 2901 R-S Test - D - 0
2072  033527  000000000000#              TSTA12          ; AL12 - 2901 S-R Test - 0 - D
2073  033530  000000000000#              TSTA13          ; AL13 - 2901 Writing/Reading Registers
2074  033531  000000000000#              TSTA14          ; AL14 - 2901 Q-Register Test
2075  033532  000000000000#              TSTA15          ; AL15 - 2901 Register Interference Test
2076  033533  000000000000#              TSTA16          ; AL16 - 2901 OR Test
2077  033534  000000000000#              TSTA17          ; AL17 - 2901 AND Test
2078  033535  000000000000#              TSTA20          ; AL20 - 2901 NOT_R_AND_S Test
2079  033536  000000000000#              TSTA21          ; AL21 - 2901 XOR Test
```

```
033537  000000000000#                   TSTA22          ; AL22 - 2901 XNOR Test
033540  000000000000#                   TSTA23          ; AL23 - 2901 R + S Test - No Carry
033541  000000000000#                   TSTA24          ; AL24 - 2901 R + S Test - With Carry
033542  000000000000#                   TSTA25          ; AL25 - 2901 R - S Test - No Carry
033543  000000000000#                   TSTA26          ; AL26 - 2901 R - S Test - With Carry
033544  000000000000#                   TSTA27          ; AL27 - 2901 S - R Test - No Carry
033545  000000000000#                   TSTA30          ; AL30 - 2901 S - R Test - With Carry
033546  000000000000#                   TSTA31          ; AL31 - 2901 Q-Register Interference Test
033547  000000000000#                   TSTA32          ; AL32 - 2901 RAM Register Interference Test
033550  000000000000#                   TSTA33          ; AL33 - Q-Register Shift
033551  000000000000#                   TSTA34          ; AL34 - Q-Register Right/Left Shift
033552  000000000000#                   TSTA35          ; AL35 - RAM Register Shift
033553  000000000000#                   TSTA36          ; AL36 - RAM Data Movement Test
033554  000000000000#                   TSTA37          ; AL37 - RAM/Q-Register Data Movement Test
033555  000000000000#                   TSTA40          ; AL40 - RAM Data Movement/Shifting Test
033556  000000000000#                   TSTA41          ; AL41 - RAM/Q-Reg Data Movement/Shifting Test
033557  000000000000#                   TSTA42          ; AL42 - 2901 Register / Constant Mux Test

                                ; MPROC Tests

033560  000000000000#           TMPROC: TSTU1           ; MP1 - CRAM Data Test RAR12=0
033561  000000000000#                   TSTU2           ; MP2 - CRAM Data Test RAR12=1
033562  000000000000#                   TSTU3           ; MP3 - CRAM Addressing Test
033563  000000000000#                   TSTU4           ; MP4 - RAR/LAR Data Path Test - LDCRAM
033564  000000000000#                   TSTU5           ; MP5 - RAR/LAR Data Path Test - RDCRAM
033565  000000000000#                   TSTU6           ; MP6 - Cram Ctrl Register Test
033566  000000000000#                   TSTU7           ; MP7 - Cram Ctrl Register Test
033567  000000000000#                   TSTU10          ; MP10 - MBUS Error Test (MPROC module only)
033570  000000000000#                   TSTU11          ; MP11 - MBUS Error Test
033571  000000000000#                   TSTU12          ; MP12 - MBUS Error Test (MPROC module only)
033572  000000000000#                   TSTU13          ; MP13 - MBUS Error Test
033573  000000000000#                   TSTU14          ; MP14 - CCCbusAvail Test
033574  000000000000#                   TSTU15          ; MP15 - CCGrntCSR Test
033575  000000000000#                   TSTU16          ; MP16 - CCFEQO Test
033576  000000000000#                   TSTU17          ; MP17 - CCCSRChng Test
033577  000000000000#                   TSTU20          ; MP20 - CCEbParErr Test
033600  000000000000#                   TSTU21          ; MP21 - CCRcvrBufAFul Test
033601  000000000000#                   TSTU22          ; MP22 - CCRcvrBufBFul Test
033602  000000000000#                   TSTU23          ; MP23 - CCXmtrAttn Test
033603  000000000000#                   TSTU24          ; MP24 - CCEbusRqst Test
033604  000000000000#                   TSTU25          ; MP25 - CCIntrActive Test
033605  000000000000#                   TSTU26          ; MP26 - CCMBSign Test
033606  000000000000#                   TSTU27          ; MP27 - CCMVParChk Test
033607  000000000000#                   TSTU30          ; MP30 - CCCbusParErr Test
033610  000000000000#                   TSTU31          ; MP31 - CCPliParErr Test
033611  000000000000#                   TSTU32          ; MP32 - CCChanErr Test
033612  000000000000#                   TSTU33          ; MP33 - CCCbLstWd Test
033613  000000000000#                   TSTU34          ; MP34 - Basic Local Store Test
033614  000000000000#                   TSTU35          ; MP35 - Local Store Data Test
033615  000000000000#                   TSTU36          ; MP36 - Local Store Address Test
033616  000000000000#                   TSTU37          ; MP37 - Local Store RAM Mode Test
033617  000000000000#                   TSTU40          ; MP40 - Jump MUX Test
033620  000000000000#                   TSTU41          ; M41 - CRAM Ctl Reg Parity Generators Test
```

```


                                    ; CBUS Tests

033621  000000000000#               TCBUS:  TSTC1           ; CB1 - Fmtr Data Loopback
033622  000000000000#                       TSTC2           ; CB2 - PLI Buffer Data Loopback
033623  000000000000#                       TSTC3           ; CB3 - CBUS to EBUF Data Transfer
033624  000000000000#                       TSTC4           ; CB4 - 2901 to CBUS Data Transfer
033625  000000000000#                       TSTC5           ; CB5 - CBUS to 2901 Multiple Word Transfer
033626  000000000000#                       TSTC6           ; CB6 - 2901 to CBUS Multiple Word Transfer
033627  000000000000#                       TSTC7           ; CB7 - CBUS to EBUS Data Transfer
033630  000000000000#                       TSTC10          ; CB10 - EBUS to CBUS Data Transfer
033631  000000000000#                       TSTC11          ; CB11 - Fmtr Cleared by 'Port Clear'
033632  000000000000#                       TSTC12          ; CB12 - Fmtr Basic Up Shifting Test
033633  000000000000#                       TSTC13          ; CB13 - Fmtr Up Shift 4 Test - BUS Ctl=Cbus
033634  000000000000#                       TSTC14          ; CB14 - Fmtr Up Shift 4 Test - BUS Ctl=Fmtr
033635  000000000000#                       TSTC15          ; CB15 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=0
033636  000000000000#                       TSTC16          ; CB16 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=0
033637  000000000000#                       TSTC17          ; CB17 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=1
033640  000000000000#                       TSTC20          ; CB20 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=1
033641  000000000000#                       TSTC21          ; CB21 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=0
033642  000000000000#                       TSTC22          ; CB22 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=0
033643  000000000000#                       TSTC23          ; CB23 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=1
033644  000000000000#                       TSTC24          ; CB24 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=1
033645  000000000000#                       TSTC25          ; CB25 - Fmtr Basic Down Shifting Test
033646  000000000000#                       TSTC26          ; CB26 - Fmtr Down Shift 4 Test
033647  000000000000#                       TSTC27          ; CB27 - Fmtr Down Shift 4 (from PLIN) Test SELRHTNIB=0
033650  000000000000#                       TSTC30          ; CB30 - Fmtr Down Shift 4 (from PLIN) Test SELRHTNIB=0
033651  000000000000#                       TSTC31          ; CB31 - Fmtr Down Shift 4 (from PLIN) Test SELRHTNIB=1
033652  000000000000#                       TSTC32          ; CB32 - Fmtr Down Shift 4 (from PLIN) Test SELRHTNIB=1
033653  000000000000#                       TSTC33          ; CB33 - Fmtr Down Shift 8 (from PLIN) Test SELRHTNIB=0
033654  000000000000#                       TSTC34          ; CB34 - Fmtr Down Shift 8 (from PLIN) Test SELRHTNIB=0
033655  000000000000#                       TSTC35          ; CB35 - Fmtr Down Shift 8 (from PLIN) Test SELRHTNIB=1
033656  000000000000#                       TSTC36          ; CB36 - Fmtr Down Shift 4 (from PLIN) Test SELRHTNIB=1
033657  000000000000#                       TSTC37          ; CB37 - Fmtr Ring Buffer Test
033660  000000000000#                       TSTC40          ; CB40 - CMUX Selected by PLINTOCMUX
033661  000000000000#                       TSTC41          ; CB41 - CBUF Loaded Correctly
033662  000000000000#                       TSTC42          ; CB42 - PMUX Selected by MPCBUFTOPLOUT
033663  000000000000#                       TSTC43          ; CB43 - PMUX Selected by MPCBUFTOPLOUT/MPZEROLFTNIB
033664  000000000000#                       TSTC44          ; CB44 - DMUX Selected Correctly
033665  000000000000#                       TSTC45          ; CB45 - CBUS Parity Checking - CBUS to CBUS Module
033666  000000000000#                       TSTC46          ; CB46 - CBUS Parity Checking - CBUS Module to CBUS
033667  000000000000#                       TSTC47          ; CB47 - CBUS Parity Checking - CBUS to CBUS Module
033670  000000000000#                       TSTC50          ; CB50 - T Field Timing Test
033671  000000000000#                       TSTC51          ; CB51 - CBUS Store
033672  000000000000#                       TSTC52          ; CB52 - CBUS Input Buffer
033673  000000000000#                       TSTC53          ; CB53 - Parity Predictor - CBIN to Fmtr/MPINDSTCOMP=0
033674  000000000000#                       TSTC54          ; CB54 - Parity Predictor - CBIN to Fmtr/MPINDSTCOMP=1
033675  000000000000#                       TSTC55          ; CB55 - Parity Predictor - Fmtr to PLOUT
033676  000000000000#                       TSTC56          ; CB56 - Parity Predictor - PLIN to Fmtr/MPCOREDUMP=0
033677  000000000000#                       TSTC57          ; CB57 - Parity Predictor - PLIN to Fmtr/MPCOREDUMP=1
033700  000000000000#                       TSTC60          ; CB60 - Parity Predictor - Fmtr to CBOUT
033701  777777  777777              TEND:   -1              ; end of tests
```

```
                                 ;#*********************************************************************
                                 ;* Test Dispatch List - in order for fault isolation
                                 ;#*********************************************************************

                                 ; Basic EBUS Interface Logic

033702  000000000000#            TDISP2: TSTE54          ; EB54 - Basic KMUX Parity
033703  000000000000#                    TSTE1           ; EB1 - CONI Works?
033704  000000000000#                    TSTE2           ; EB2 - RH20/IPA Interaction
033705  000000000000#                    TSTE3           ; EB3 - RH20/IPA Interaction
033706  000000000000#                    TSTE4           ; EB4 - RH20/IPA Interaction
033707  000000000000#                    TSTE5           ; EB5 - RH20/IPA Interaction

                                 ; CSR Register Logic

033710  000000000000#                    TSTE6           ; EB6 - CSR Cleared After Reset
033711  000000000000#                    TSTE7           ; EB7 - Read/Write CSR Bits
033712  000000000000#                    TSTE10          ; EB10 - CSR Bits Interaction

                                 ; EBUF Logic

033713  000000000000#                    TSTE11          ; EB11 - EBUF Access Test
033714  000000000000#                    TSTE12          ; EB12 - EBUF/CSR Conflict Test
033715  000000000000#                    TSTE14          ; EB14 - EBUF/CSR Conflict Test
033716  000000000000#                    TSTE15          ; EB15 - EBUF Reset Test
033717  000000000000#                    TSTE16          ; EB16 - EBUF Data Path Test

                                 ; Reset Logic

033720  000000000000#                    TSTE17          ; EB17 - CSR Bit 18 Self Clearing Test
033721  000000000000#                    TSTE20          ; EB20 - CSR Bit 18 Reset Test
033722  000000000000#                    TSTE21          ; EB21 - CSR Bit 18 Reset Test

                                 ; EBUS Parity Logic

033723  000000000000#                    TSTE23          ; EB23 - Par Gen - Not Forcing an Error
033724  000000000000#                    TSTE24          ; EB24 - Par Gen - Forcing an Error

                                 ; Basic CRAM logic

033725  000000000000#                    TSTU1           ; MP1 - CRAM Data Test RAR12=0
033726  000000000000#                    TSTU2           ; MP2 - CRAM Data Test RAR12=1
033727  000000000000#                    TSTU3           ; MP3 - CRAM Addressing Test
033730  000000000000#                    TSTU4           ; MP4 - RAR/LAR Data Path Test - LDCRAM
033731  000000000000#                    TSTU5           ; MP5 - RAR/LAR Data Path Test - RDCRAM

                                 ; Run/Single Step Logic

033732  000000000000#                    TSTE53          ; EB53 - MPROC Run Bit
033733  000000000000#                    TSTE57          ; EB57 - Verify 'Diag Single Cyc'
```

```

                                        ; Addl CRAM tests

033734  000000000000#                           TSTU6           ; MP6 - Cram Ctrl Register Test
033735  000000000000#                           TSTU7           ; MP7 - Cram Ctrl Register Test
033736  000000000000#                           TSTU41          ; M41 - CRAM Ctl Reg Parity Generators Test

                                        ; Sequencer Tests

033737  000000000000#                           TSTS1           ; SE1 - JMAP Instruction
033740  000000000000#                           TSTS2           ; SE2 - JZ Instruction
033741  000000000000#                           TSTS3           ; SE3 - JZ Instruction - Effect on Reg/Ctr
033742  000000000000#                           TSTS4           ; SE4 - JZ Instruction - Effect on Reg/Ctr
033743  000000000000#                           TSTS5           ; SE5 - CONT Instruction
033744  000000000000#                           TSTS6           ; SE6 - CJP Instruction
033745  000000000000#                           TSTS7           ; SE7 - CJV Instruction
033746  000000000000#                           TSTS10          ; SE10 - CJS Instruction
033747  000000000000#                           TSTS11          ; SE11 - CJS Instruction
033750  000000000000#                           TSTS12          ; SE12 - CJS Instruction
033751  000000000000#                           TSTS13          ; SE13 - CRTN Instruction
033752  000000000000#                           TSTS14          ; SE14 - CRTN Instruction
033753  000000000000#                           TSTS15          ; SE15 - Stack location #1
033754  000000000000#                           TSTS16          ; SE16 - Stack location #2
033755  000000000000#                           TSTS17          ; SE17 - Stack location #3
033756  000000000000#                           TSTS20          ; SE20 - Stack location #4
033757  000000000000#                           TSTS21          ; SE21 - Stack location #5
033760  000000000000#                           TSTS22          ; SE22 - Increment/Decrement Stack Pointer
033761  000000000000#                           TSTS23          ; SE23 - Increment/Decrement Stack Pointer
033762  000000000000#                           TSTS24          ; SE24 - Increment/Decrement Stack Pointer
033763  000000000000#                           TSTS25          ; SE25 - Increment/Decrement Stack Pointer
033764  000000000000#                           TSTS26          ; SE26 - Increment/Decrement Stack Pointer
033765  000000000000#                           TSTS27          ; SE27 - Overflowing the Stack
033766  000000000000#                           TSTS30          ; SE30 - CJPP Instruction
033767  000000000000#                           TSTS31          ; SE31 - CJPP Instruction
033770  000000000000#                           TSTS32          ; SE32 - CJPP Instruction
033771  000000000000#                           TSTS33          ; SE33 - LDCT Instruction
033772  000000000000#                           TSTS34          ; SE34 - LDCT Instruction
033773  000000000000#                           TSTS35          ; SE35 - Register/Counter
033774  000000000000#                           TSTS36          ; SE36 - JRP Instruction
033775  000000000000#                           TSTS37          ; SE37 - JSRP Instruction
033776  000000000000#                           TSTS40          ; SE40 - JSRP Instruction
033777  000000000000#                           TSTS41          ; SE41 - JSRP Instruction
034000  000000000000#                           TSTS42          ; SE42 - PUSH Instruction
034001  000000000000#                           TSTS43          ; SE43 - PUSH Instruction
034002  000000000000#                           TSTS44          ; SE44 - PUSH Instruction
```

```
034003  000000000000#          TSTS45          ; SE45 - LOOP Instruction
034004  000000000000#          TSTS46          ; SE46 - LOOP Instruction
034005  000000000000#          TSTS47          ; SE47 - LOOP Instruction
034006  000000000000#          TSTS50          ; SE50 - RFCT Instruction
034007  000000000000#          TSTS51          ; SE51 - RFCT Instruction
034010  000000000000#          TSTS52          ; SE52 - RFCT Instruction
034011  000000000000#          TSTS53          ; SE53 - RFCT Instruction
034012  000000000000#          TSTS54          ; SE54 - RPCT Instruction
034013  000000000000#          TSTS55          ; SE55 - RPCT Instruction
034014  000000000000#          TSTS56          ; SE56 - TWB Instruction
034015  000000000000#          TSTS57          ; SE57 - TWB Instruction
034016  000000000000#          TSTS60          ; SE60 - TWB Instruction
034017  000000000000#          TSTS61          ; SE61 - TWB Instruction
034020  000000000000#          TSTS62          ; SE62 - TWB Instruction
034021  000000000000#          TSTS63          ; SE63 - TWB Instruction
034022  000000000000#          TSTS64          ; SE64 - Stack Interference Tests
034023  000000000000#          TSTS65          ; SE65 - Stack Interference Tests
034024  000000000000#          TSTS66          ; SE66 - Reg/Ctr Interference Tests
034025  000000000000#          TSTS67          ; SE67 - Reg/Ctr Interference Tests
034026  000000000000#          TSTS70          ; SE70 - Full Speed Sequencer Test

                       ; Miscellaneous

034027  000000000000#          TSTE13          ; EB13 - KMUX Parity Generators
034030  000000000000#          TSTE35          ; EB35 - Constant MUX
034031  000000000000#          TSTE36          ; EB36 - Constant MUX Interference
034032  000000000000#          TSTE37          ; EB37 - CSR Bit 24 - EBUS PE Occurs at Wrong Time

                       ; 2901 Tests

034033  000000000000#          TSTA1           ; AL1 - 2901 0's Test
034034  000000000000#          TSTA2           ; AL2 - 2901 1's Test
034035  000000000000#          TSTA3           ; AL3 - 2901 OR Test - D OR 0
034036  000000000000#          TSTA4           ; AL4 - 2901 AND Test - D AND 0
034037  000000000000#          TSTA5           ; AL5 - 2901 NOT_R_AND_S Test - D NOT_RS 0
034040  000000000000#          TSTA6           ; AL6 - 2901 XOR Test - D XOR 0
034041  000000000000#          TSTA7           ; AL7 - 2901 XNOR Test - D XNOR 0
034042  000000000000#          TSTA10          ; AL10 - 2901 R+S Test - D + 0
034043  000000000000#          TSTA11          ; AL11 - 2901 R-S Test - D - 0
034044  000000000000#          TSTA12          ; AL12 - 2901 S-R Test - 0 - D
034045  000000000000#          TSTA13          ; AL13 - 2901 Writing/Reading Registers
034046  000000000000#          TSTA14          ; AL14 - 2901 Q-Register Test
034047  000000000000#          TSTA15          ; AL15 - 2901 Register Interference Test
034050  000000000000#          TSTA16          ; AL16 - 2901 OR Test
034051  000000000000#          TSTA17          ; AL17 - 2901 AND Test
034052  000000000000#          TSTA20          ; AL20 - 2901 NOT_R_AND_S Test
034053  000000000000#          TSTA21          ; AL21 - 2901 XOR Test
034054  000000000000#          TSTA22          ; AL22 - 2901 XNOR Test
034055  000000000000#          TSTA23          ; AL23 - 2901 R + S Test - No Carry
034056  000000000000#          TSTA24          ; AL24 - 2901 R + S Test - With Carry
034057  000000000000#          TSTA25          ; AL25 - 2901 R - S Test - No Carry
```

```
 2332  034060  000000000000#               TSTA26          ; AL26 - 2901 R - S Test - With Carry
 2333  034061  000000000000#               TSTA27          ; AL27 - 2901 S - R Test - No Carry
 2334  034062  000000000000#               TSTA30          ; AL30 - 2901 S - R Test - With Carry
 2335  034063  000000000000#               TSTA31          ; AL31 - 2901 Q-Register Interference Test
 2336  034064  000000000000#               TSTA32          ; AL32 - 2901 RAM Register Interference Test
 2337  034065  000000000000#               TSTA33          ; AL33 - Q-Register Shift
 2338  034066  000000000000#               TSTA34          ; AL34 - Q-Register Right/Left Shift
 2339  034067  000000000000#               TSTA35          ; AL35 - RAM Register Shift
 2340  034070  000000000000#               TSTA36          ; AL36 - RAM Data Movement Test
 2341  034071  000000000000#               TSTA37          ; AL37 - RAM/Q-Register Data Movement Test
 2342  034072  000000000000#               TSTA40          ; AL40 - RAM Data Movement/Shifting Test
 2343  034073  000000000000#               TSTA41          ; AL41 - RAM/Q-Reg Data Movement/Shifting Test
 2344  034074  000000000000#               TSTA42          ; AL42 - 2901 Register / Constant Mux Test
 2345
 2346                                ; Miscellaneous Tests
 2347
 2348  034075  000000000000#               TSTE40          ; EB40 - EBUS Transfer via DATAO's
 2349  034076  000000000000#               TSTE41          ; EB41 - EBUS Transfer via DATAI's
 2350
 2351                                ; CSR Register Logic
 2352
 2353  034077  000000000000#               TSTE22          ; EB22 - Read/Write EBUF When Ucode Running
 2354  034100  000000000000#               TSTE25          ; EB25 - Read only CSR Bits
 2355  034101  000000000000#               TSTE26          ; EB26 - CSR 01 'DIAG RQST CSR'
 2356  034102  000000000000#               TSTE27          ; EB27 - CSR 01 'DIAG RQST CSR'
 2357  034103  000000000000#               TSTE30          ; EB30 - CSR 01 'DIAG RQST CSR'
 2358  034104  000000000000#               TSTE31          ; EB31 - CSR 01 'DIAG RQST CSR'
 2359  034105  000000000000#               TSTE32          ; EB32 - CSR 01 'DIAG RQST CSR'
 2360  034106  000000000000#               TSTE33          ; EB33 - CSR 01 'DIAG RQST CSR'
 2361  034107  000000000000#               TSTE34          ; EB34 - CSR 02 'DIAG CSR CHNG'
 2362  034110  000000000000#               TSTE42          ; EB42 - CSR 05 'Rqst Interrupt'
 2363  034111  000000000000#               TSTE43          ; EB43 - CSR 05 'Rqst Interrupt'
 2364  034112  000000000000#               TSTE44          ; EB44 - CSR 05 'Rqst Interrupt'
 2365  034113  000000000000#               TSTE45          ; EB45 - CSR 06 'CRAM Par Err'
 2366  034114  000000000000#               TSTE46          ; EB46 - CSR 07 'MBUS Err'
 2367  034115  000000000000#               TSTE47          ; EB47 - CSR 11 'Spare' Bit
 2368  034116  000000000000#               TSTE50          ; EB50 - CSR 12 'Disable Complete' Bit
 2369  034117  000000000000#               TSTE51          ; EB51 - CSR 13 'Enable Complete' Bit
 2370  034120  000000000000#               TSTE73          ; EB73 - CSR 25-26,28 KL side
 2371  034121  000000000000#               TSTE74          ; EB74 - CSR 25-26,28 Port side
 2372  034122  000000000000#               TSTE75          ; EB75 - CSR 27,30-31 KL side
 2373  034123  000000000000#               TSTE76          ; EB76 - CSR 27,30-31 Port side
 2374  034124  000000000000#               TSTE55          ; EB55 - Clearing MPROC Run Bit
 2375  034125  000000000000#               TSTE56          ; EB56 - Clearing MPROC Run Bit
 2376
 2377                                ; MBUS Error Logic
 2378
 2379  034126  000000000000#               TSTU10          ; MP10 - MBUS Error Test (MPROC module only)
 2380  034127  000000000000#               TSTU11          ; MP11 - MBUS Error Test
 2381  034130  000000000000#               TSTU12          ; MP12 - MBUS Error Test (MPROC module only)
 2382  034131  000000000000#               TSTU13          ; MP13 - MBUS Error Test
```

```

                                ; Local Storage Tests

034132  000000000000#                   TSTU34          ; MP34 - Basic Local Store Test
034133  000000000000#                   TSTU35          ; MP35 - Local Store Data Test
034134  000000000000#                   TSTU36          ; MP36 - Local Store Address Test
034135  000000000000#                   TSTU37          ; MP3  - Local Store RAM Mode Test

                                ; Miscellaneous

034136  000000000000#                   TSTU40          ; MP40 - Jump MUX Test

                                ; Interrupt Logic

034137  000000000000#                   TSTE60          ; EB60 - PI Level n/0 Interrupts
034140  000000000000#                   TSTE61          ; EB61 - Check PIA Level 0
034141  000000000000#                   TSTE62          ; EB62 - Check Interrupt on Level 1-7
034142  000000000000#                   TSTE63          ; EB63 - CSR 05 'Rqst Interrupt' Clearing
034143  000000000000#                   TSTE64          ; EB64 - IOP Function 0 - 40+2N Interrupt
034144  000000000000#                   TSTE65          ; EB65 - IOP Function 1 - 40+2N Interrupt
034145  000000000000#                   TSTE70          ; EB70 - IOP Function 4 - Examine
034146  000000000000#                   TSTE71          ; EB71 - IOP Function 5 - Deposit
034147  000000000000#                   TSTE72          ; EB72 - IOP Function 7 - Examine/Increment

                                ; Basic CBUS Tests

034150  000000000000#                   TSTC1           ; CB1 - Fmtr Data Loopback
034151  000000000000#                   TSTC2           ; CB2 - PLI Buffer Data Loopback
034152  000000000000#                   TSTC11          ; CB11 - Fmtr Cleared by 'Port Clear'

                                ; Condition Code Tests

034153  000000000000#                   TSTU14          ; MP14 - CCCbusAvail Test
034154  000000000000#                   TSTU15          ; MP15 - CCGrntCSR Test
034155  000000000000#                   TSTU16          ; MP16 - CCFEQ0 Test
034156  000000000000#                   TSTU17          ; MP17 - CCCSRChng Test
034157  000000000000#                   TSTU20          ; MP20 - CCEbParErr Test
034160  000000000000#                   TSTU21          ; MP21 - CCRcvrBufAFul Test
034161  000000000000#                   TSTU22          ; MP22 - CCRcvrBufBFul Test
034162  000000000000#                   TSTU23          ; MP23 - CCXmtrAttn Test
034163  000000000000#                   TSTU24          ; MP24 - CCEbusRqst Test
034164  000000000000#                   TSTU25          ; MP25 - CCIntrActive Test
034165  000000000000#                   TSTU26          ; MP26 - CCMBSign Test
034166  000000000000#                   TSTU27          ; MP27 - CCMVParChk Test
034167  000000000000#                   TSTU30          ; MP30 - CCCbusParErr Test
034170  000000000000#                   TSTU31          ; MP31 - CCPLiParErr Test
034171  000000000000#                   TSTU32          ; MP32 - CCChanErr Test
034172  000000000000#                   TSTU33          ; MP33 - CCCbLstWd Test

                                ; CBUS Data Transfer

034173  000000000000#                   TSTC3           ; CB3 - CBUS to EBUF Data Transfer
```

```
034174  000000000000#                   TSTC4           ; CB4 - 2901 to CBUS Data Transfer
034175  000000000000#                   TSTC5           ; CB5 - CBUS to 2901 Multiple Word Transfer
034176  000000000000#                   TSTC6           ; CB6 - 2901 to CBUS Multiple Word Transfer
034177  000000000000#                   TSTC7           ; CB7 - CBUS to EBUS Data Transfer
034200  000000000000#                   TSTC10          ; CB10 - EBUS to CBUS Data Transfer

                                ; Formatter Tests

034201  000000000000#                   TSTC12          ; CB12 - Fmtr Basic Up Shifting Test
034202  000000000000#                   TSTC13          ; CB13 - Fmtr Up Shift 4 Test - BUS Ctl=Cbus
034203  000000000000#                   TSTC14          ; CB14 - Fmtr Up Shift 4 Test - BUS Ctl=Fmtr
034204  000000000000#                   TSTC15          ; CB15 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=0
034205  000000000000#                   TSTC16          ; CB16 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=0
034206  000000000000#                   TSTC17          ; CB17 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=1
034207  000000000000#                   TSTC20          ; CB20 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=1
034210  000000000000#                   TSTC21          ; CB21 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=0
034211  000000000000#                   TSTC22          ; CB22 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=0
034212  000000000000#                   TSTC23          ; CB23 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=1
034213  000000000000#                   TSTC24          ; CB24 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=1
034214  000000000000#                   TSTC25          ; CB25 - Fmtr Basic Down Shifting Test
034215  000000000000#                   TSTC26          ; CB26 - Fmtr Down Shift 4 Test
034216  000000000000#                   TSTC27          ; CB27 - Fmtr Down Shift 4 (from PLIN) Test SELRHTNIB=0
034217  000000000000#                   TSTC30          ; CB30 - Fmtr Down Shift 4 (from PLIN) Test SELRHTNIB=0
034220  000000000000#                   TSTC31          ; CB31 - Fmtr Down Shift 4 (from PLIN) Test SELRHTNIB=1
034221  000000000000#                   TSTC32          ; CB32 - Fmtr Down Shift 4 (from PLIN) Test SELRHTNIB=1
034222  000000000000#                   TSTC33          ; CB33 - Fmtr Down Shift 8 (from PLIN) Test SELRHTNIB=0
034223  000000000000#                   TSTC34          ; CB34 - Fmtr Down Shift 8 (from PLIN) Test SELRHTNIB=0
034224  000000000000#                   TSTC35          ; CB35 - Fmtr Down Shift 8 (from PLIN) Test SELRHTNIB=1
034225  000000000000#                   TSTC36          ; CB36 - Fmtr Down Shift 4 (from PLIN) Test SELRHTNIB=1
034226  000000000000#                   TSTC37          ; CB37 - Fmtr Ring Buffer Test

                                ; Miscellaneous CBUS module tests

034227  000000000000#                   TSTC40          ; CB40 - CMUX Selected by PLINTOCMUX
034230  000000000000#                   TSTC41          ; CB41 - CBUF Loaded Correctly
034231  000000000000#                   TSTC42          ; CB42 - PMUX Selected by MPCBUFTOPLOUT
034232  000000000000#                   TSTC43          ; CB43 - PMUX Selected by MPCBUFTOPLOUT/MPZEROLFTNIB
034233  000000000000#                   TSTC44          ; CB44 - DMUX Selected Correctly
034234  000000000000#                   TSTC45          ; CB45 - CBUS Parity Checking - CBUS to CBUS Module
034235  000000000000#                   TSTC46          ; CB46 - CBUS Parity Checking - CBUS Module to CBUS
034236  000000000000#                   TSTC47          ; CB47 - CBUS Parity Checking - CBUS to CBUS Module
034237  000000000000#                   TSTC50          ; CB50 - T Field Timing Test
034240  000000000000#                   TSTC51          ; CB51 - CBUS Store
034241  000000000000#                   TSTC52          ; CB52 - CBUS Input Buffer
034242  000000000000#                   TSTC53          ; CB53 - Parity Predictor - CBIN to Fmtr/MPINDSTCOMP=0
034243  000000000000#                   TSTC54          ; CB54 - Parity Predictor - CBIN to Fmtr/MPINDSTCOMP=1
034244  000000000000#                   TSTC55          ; CB55 - Parity Predictor - Fmtr to PLOUT
034245  000000000000#                   TSTC56          ; CB56 - Parity Predictor - PLIN to Fmtr/MPCOREDUMP=0
034246  000000000000#                   TSTC57          ; CB57 - Parity Predictor - PLIN to Fmtr/MPCOREDUMP=1
034247  000000000000#                   TSTC60          ; CB60 - Parity Predictor - Fmtr to CBOUT
034250  777777  777777                  -1              ; end of tests
```

```
                                  ;#*******************************************************************
                                  ;* FLTHAN - Fault handler
                                  ;
                                  ; AC3 - Address of module list (0 if none, last entry has bit 0 set)
                                  ; AC4 - Weight to be added to probability list
                                  ;#*******************************************************************

034251  322 03 0 00 052526        FLTHAN: JUMPE   3,[RTN]                      ; nothing? yes - return
034252  261 17 0 00 000001                PUT     1                            ; save AC1
034253  550 01 0 03 000000        FLTHA0: HRRZ    1,(3)                        ; get module entry
034254  272 04 0 01 031752                ADDM    4,NETBAD-1(1)                ; update module probability count
034255  377 00 0 00 000004                SOSG    4                            ; adjust weight - less than 1?
034256  201 04 0 00 000001                MOVEI   4,1                          ; yes - ensure at least 1
034257  332 00 0 00 064606'               SKIPE   IDEBUG                       ; fault isolation debug mode?
                                          GO      [RPUT   (0,1)                ; yes - print it out
                                                   HRRZ   1,NETDES-1(1)
                                                   MOVE   1,@1
                                                   LDB    [POINT 7,1,20]
                                                   CAIN   72
                                                   AND    1,[777760,,077777]
                                                   LDB    [POINT 7,1,27]
                                                   CAIN   72
                                                   AND    1,[777777,,700377]
                                                   MOVEI  1
                                                   PNTAL
                                                   PSP
                                                   RGET   (1,0)
034260  260 17 0 00 055413                         RTN]
034261  200 01 0 03 000000                MOVE    1,(3)                        ; get module entry
034262  350 00 0 00 000003                AOS     3                            ; point to next module entry
034263  607 01 0 00 400000                TLNN    1,400000                     ; last one?
034264  254 00 0 00 034253                JRST    FLTHA0                       ; no - loop till done
034265  262 17 0 00 000001                GET     1                            ; restore AC1
034266  263 17 0 00 000000                RTN                                  ; return
```

```
                                  ;#*********************************************************************
                                  ;* FLTCHK - See if this test should fail based on possible modules list
                                  ;#*********************************************************************

034267  261 17 0 00 000001        FLTCHK: PUT     1                       ; save AC1

                                  ; Check high probability modules

034270  200 03 0 00 047024                MOVE    3,TSTHIG                ; get addr of high prob module list
034271  322 03 0 00 034304                JUMPE   3,FLTCH1                ; nothing? yes - try low probability
034272  550 01 0 03 000000        FLTCH0: HRRZ    1,(3)                   ; get module entry
034273  316 01 0 00 047032                CAMN    1,FLTNUM                ; module matches fault?
034274  474 15 0 00 000000                SETO    ERFLG,                  ; yes - set error flag
034275  200 01 0 03 000000                MOVE    1,(3)                   ; get module entry
034276  350 00 0 00 000003                AOS     3                       ; point to next module entry
034277  607 01 0 00 400000                TLNN    1,400000                ; last one?
034300  254 00 0 00 034272                JRST    FLTCH0                  ; no - loop till done
034301  326 15 0 00 034313                JUMPN   ERFLG,FLTCHX            ; error flag set? yes - exit

                                  ; Check low probability modules

034302  200 03 0 00 047025                MOVE    3,TSTLOW                ; get addr of low prob module list
034303  322 03 0 00 034313                JUMPE   3,FLTCHX                ; nothing? yes - exit
034304  550 01 0 03 000000        FLTCH1: HRRZ    1,(3)                   ; get module entry
034305  316 01 0 00 047032                CAMN    1,FLTNUM                ; module matches fault?
034306  474 15 0 00 000000                SETO    ERFLG,                  ; yes - set error flag
034307  200 01 0 03 000000                MOVE    1,(3)                   ; get module entry
034310  350 00 0 00 000003                AOS     3                       ; point to next module entry
034311  607 01 0 00 400000                TLNN    1,400000                ; last one?
034312  254 00 0 00 034304                JRST    FLTCH1                  ; no - loop till done

                                  ; Exit

034313  262 17 0 00 000001        FLTCHX: GET     1                       ; restore AC1
034314  263 17 0 00 000000                RTN                             ; return
```

```
                                        ;#*****************************************************************
                                        ;* TSTDEC - Decode test specified as XXnnn where XX is the test class
                                        ;*          and nnn is the test number.
                                        ;
                                        ; Argument:     AC1 - Contains SIXBIT test specification
                                        ;
                                        ; Returns:      +1 - Test not recognized (AC1 not preserved).
                                        ;               +2 - Test number out of range (AC1 contains maximum
                                        ;                    test number.
                                        ;               +3 - Ok, AC1 contains test number decoded.  Also,
                                        ;                    TSTCLS, TSTNUM, TSTADD are set up.
                                        ;#*****************************************************************

034315  261 17 0 00 000000              TSTDEC: RPUT    (0,2,3)                 ; save AC's
034320  200 03 0 00 000001                      MOVE    3,1                     ; save AC1

                                        ; Decode the 2 character test type

034321  135 01 0 00 055433                      LDB     1,[POINT 12,3,11]       ; get first 2 characters
034322  242 01 0 00 000030                      LSH     1,^D24                  ; position correctly
034323  201 02 0 00 031275                      MOVEI   2,INDLIS                ; get address of individual test list
034324  260 17 0 00 032562*                     GO      .COMM                   ; handle command decoding
034325  254 00 0 00 034354                       JRST   TSTDED                  ; check if only 1 character
034326  202 01 0 00 047010                      MOVEM   1,TSTCLS                ; save test class

                                        ; Decode the test number

034327  135 01 0 00 055434                      LDB     1,[POINT 24,3,35]       ; get test number
034330  242 01 0 00 000014                      LSH     1,^D12                  ; left justify
034331  260 17 0 00 000000*             TSTDEO: GO      CONVSX                  ; convert to octal
034332  254 00 0 00 034350                       JRST   TSTDEE                  ; error - continue
034333  202 01 0 00 047011                      MOVEM   1,TSTNUM                ; save test number
034334  200 02 0 00 047010                      MOVE    2,TSTCLS                ; get test class
                                                XCT     [MOVEI  2,TSEQ-TEBUS    ; get maximum tests in the
                                                         MOVEI  2,TALU-TSEQ     ;   particular category
                                                         MOVEI  2,TMPROC-TALU
                                                         MOVEI  2,TCBUS-TMPROC
034335  256 00 0 02 055435                               MOVEI  2,TEND-TCBUS](2)
034336  303 01 0 00 000000                      CAILE   1,0                     ; test number in the range
034337  313 01 0 00 000002                      CAMLE   1,2                     ;   1..n?
                                                JRST    [MOVE   1,2             ; no - get maximum test number
034340  254 00 0 00 055442                               JRST   TSTDER]         ;   and exit
```

```

                                  ; Succeeded, also set up TSTADD

034341  200 02 0 00 047010                MOVE    2,TSTCLS                ; get test class
                                          XCT     [MOVEI  2,TEBUS         ; get starting address in address
                                                   MOVEI  2,TSEQ          ;   table
                                                   MOVEI  2,TALU
                                                   MOVEI  2,TMPROC
034342  256 00 0 02 054210                         MOVEI  2,TCBUS](2)
034343  270 02 0 00 000001                ADD     2,1                     ; include the test number
034344  370 00 0 00 000002                SOS     2                       ; allow for tests 1..
034345  202 02 0 00 047015                MOVEM   2,TSTADD                ; save the dispatch table address

                                  ; Exit

034346  350 00 0 17 777775                AOS     -3(P)                   ; yes - set up RTN+3
034347  350 00 0 17 777775        TSTDER: AOS     -3(P)                   ; set up RTN+2
034350  262 17 0 00 000003        TSTDEE: RGET    (3,2,0)                 ; error return (nothing recognized)
034353  263 17 0 00 000000                RTN                             ; return

                                  ; Command not recognized, maybe only one alphabetic character was given

034354  135 01 0 00 055444        TSTDED: LDB     1,[POINT 6,3,5]         ; get first character
034355  242 01 0 00 000036                LSH     1,^D30                  ; position correctly
034356  201 02 0 00 031275                MOVEI   2,INDLIS                ; get address of individual test list
034357  260 17 0 00 034324*               GO      .COMM                   ; handle command decoding
034360  254 00 0 00 034350                 JRST   TSTDEE                  ; error return
034361  202 01 0 00 047010                MOVEM   1,TSTCLS                ; save test class
034362  135 01 0 00 055445                LDB     1,[POINT 30,3,35]       ; get test number
034363  242 01 0 00 000006                LSH     1,6                     ; left justify
034364  254 00 0 00 034331                JRST    TSTDEO                  ; continue

```

```
        SUBTTL  Port Debugger

;#*************************************************************************
;* TSTDEB - Console command utility for the port.
;
; Command               Action
; -------               ------
;
;  ?                Print list of commands available.
;
;  HELP             Print list of commands and description.
;  EXIT             Exit debugger mode and go back to 'What Test' level.
;  DDT              Enter DDT.
;
;  TAKE             Take commands from specified file
;
;  CONFIG           Determine/print configuration
;  CONPNT           Print configuration
;  SELECT           Select device to test
;
;  SET ADDR adr             Set starting address for a START/CONT command.
;      CSR arg              Set starting data to be written to CSR for START/CONT.
;      PARITY arg           Set automatic parity generation flag when loading cram.
;      EBUF arg             Set preserve EBUF flag for single stepping.
;      LENGTH arg           Set length of data transfer operations.
;      DATA arg             Set data pattern to use for data transfer operations.
;      FNAME name ext       Set default microcode file name.
;      WHAT                 Print current selections.
;      PATLIS               Print data patterns.
;
;  ECSR             Print contents of the CSR register (in octal).
;  DCSR arg         Change the contents of the CSR registers.
;  ZCSR             Clear CSR register.
;
;  RESET            Issue an EBUS reset.
;  CLEAR            Issue a 'Clear Port'.
;
;  START adr        Start port microcode by setting CSR Bit 21 'MPROC Run'.
;                   If address is supplied, start at that address; if no
;                   address is supplied, start at location 0.
;
;  STOP             Stop port microcode by clearing CSR Bit 21 'MPROC Run'.
;                   Report the last CRAM address also.
;
;  CONT             Clear the single step bit and set 'MPROC Run' in the port.
;
;  SSTEP arg        Single step the port microcode 'ARG' times.  Report
;                   final CRAM address.
;
;  STRACE arg       Single step the port microcode 'ARG' times.  Report
;                   the CRAM address and CSR register contents after each
;                   micro-instruction.
;  Response          Action
;  --------          ------
;
;  SPRINT arg       Print single step history data, 'ARG' entries.
;  SCLEAR           Clear single step history data.
;  SSINIT           Initialize single step data
;
;  EEBUF            Print contents of EBUF register.
;  DEBUF arg        Deposit data to EBUF register.
;  ZEBUF            Zero EBUF register.
;
;  ELAR             Print contents of LAR register.
;  DRAR arg         Deposit data to RAR register.
;  ZRAR             Zero RAR register.
;
;  DCRAM adr,adr    Deposit data into port CRAM addresses specified by 'adr1'
;                   and 'adr2'.  The data will be queried for each address.
;                   The port microcode must be stopped.
;
;  BCRAM adr,adr    Deposit data as above, data is queried by bit field.
;
;  ACRAM adr,adr    Alter CRAM locations, addresses selected as above.
;
;  ECRAM adr,adr    Examine port CRAM addresses.  'Adr1' and 'adr2' specify
;                   examine limits.  The port microcode must be stopped.
;
;  LCRAM adr,adr    List port CRAM addresses.  'Adr1' and 'adr2' specify
;                   examine limits.  The port microcode must be stopped.
;
;  ZCRAM adr,adr    Zero port CRAM addresses.  'Adr1' and 'adr2' specify
;                   examine limits.  The port microcode must be stopped.
;
;  ELOCS adr,adr    Examine local storage contents
;  DLOCS adr,data   Deposit local storage contents
;  ZLOCS adr,adr    Zero local storage contents
;
;  E2901 adr,adr    Examine 2901 registers
;  D2901 adr,data   Deposit 2901 registers
;  Z2901 adr,adr    Zero 2901 registers
;
;  LCONI            Loop on CONI
;  LCONO            Loop on CONO
;  LDATAI           Loop on DATAI
;  LDATAO           Loop on DATAO
;  LROUTN           Loop on user supplied routine at DROUTN:
;
;  MARK adr..       Set mark bit in CRAM locations
;  RMARK adr..      Remove mark bit at CRAM address 'adr'
;  CMARK adr,adr    Clear mark bit in CRAM locations
;  LMARK            List locations with mark bit set
;  Response            Action
;  --------            ------
;
;  BREAK adr..     Set a breakpoint at CRAM address
;  RBREAK adr..    Remove breakpoint at CRAM address 'adr'
;  CBREAK adr,adr  Clear all breakpoints
;  LBREAK adr,adr  List breakpoints
;
;  LOAD arg        Load microcode into CRAM from a test.  'Arg' specifies
;                  test number.
;  LIST arg        List microcode of a test.  'Arg' specifies test number.
;  FLOAD           Load microcode from disk file into CRAM.
;  FVERFY          Verify microcode loaded.
;  FLIST           List microcode on disk file.
;  FEXAM           Examine microcode on disk file.
;
;  TRANSL arg      Translate given CSR bits to English.
;  CCODE          Print current status of condition codes.
;  ESTACK         Examine 2910 stack contents
;
;  FILLNX arg      Fill CRAM with data as 'J=.+1,CTL=x', where 'x' is given.
;  FILLPC arg      Fill CRAM with data as 'J=.,CTL=x', where 'x' is given.
;
;  IPRINT          Print interrupt history
;  ICLEAR          Clear interrupt history
;  IINIT           Initialize interrupt system
;
;  RDINIT          Initialize e/g for a read
;  WRINIT          Initialize e/g for a write
;  BPRINT          Print buffer contents
;  CCWPNT          Print CCW list
;  LOGPNT          Print logout data
;  COMPAR          Compare data buffer contents
;
;  Switches:  To complement switch, type switch name
;
;  SWITCH          Enter switches
;  SWPRIN          Print current switches
;
;  NOPNT    LOOPER  RELIAB   TRACE   LOOPGM  DSPEAR  SDEBUG  NIPORT  MMPROC
;  PNTLPT   ERSTOP  TXTINH   INHFLT  LOOPTS  UDEBUG  LDEBUG  CIPORT  MCBUS
;  DING     PALERS  OPRSEL   INHMSG  RUNALL  MDEBUG  IDEBUG
```

```
                                        ; Common Error Messages For 'DEBUG' Mode:
                                        ;
                                        ; "? Missing argument" - Response to "DEBUG>" prompt expects a command
                                        ;       plus an argument.
                                        ;
                                        ; "? Argument Error" - An attempt was made to decode an octal or decimal
                                        ;       argument.  It was undecipherable.
                                        ;
                                        ; "? Range nonsensical" - The beginning address is greater than ending
                                        ;       address.
                                        ;
                                        ; "? Range of CRAM locations is 0000-7777" - Argument(s) typed were not
                                        ;       it range.
                                        ;
                                        ; "? Port not stopped" - An operation requiring the port to be stopped
                                        ;       was issued when 'MPROC Run' was set.
                                        ;
                                        ; "? Port not running" - A function was issued to the port when it was
                                        ;       stopped.  It should have been running.
                                        ;
                                        ; "? Test number out of range" - In response to the Test Number question.
                                        ;
                                        ; "? Error accessing CSR registers" - A read of CSR register failed.
                                        ;
                                        ; "? CSR has MBUS Error bit set - must do a reset first" - A operation
                                        ;       requiring ability to write CSR was done, but MBUS ERROR bit
                                        ;       is set.
                                        ;
                                        ; "? Test specified does not have microcode" - An attempt to load ucode
                                        ;       from this test failed because the test doesn't have any.
                                        ;
                                        ; "? Unrecognized command" - Unrecognized command entered.
                                        ;#*************************************************************************

        034365  200 01 0 00 055451      TSTDEB: MOVE    1,[FMSGCD (DEBUG <CR>)]
        034366  260 17 0 00 032705*             GO      CHKARG                  ; check for argument
        034367  263 17 0 00 000000              RTN                             ; error - exit
        034370  202 17 0 00 064635'             MOVEM   P,SAVCMD#               ; save stack pointer

                                        ; Determine port under test

        034371  260 17 0 00 032520      CMDO:   GO      PTEST                   ; determine port/set up MBCN
        034372  260 17 0 00 032537              GO      PTESTP                  ; print results
        034373  322 16 0 00 052526              JUMPE   MBCN,[RTN]              ; exit if no ports are selected
        034374  260 17 0 00 033171*             GO      DEVREQ                  ; request the device
                                                JRST    [FMSGCD <[? Configuration changed - reconfigure using the CONFIG command]>
        034375  254 00 0 00 055470                       RTN]                           ; failed - return
```

```

                                  ; Initialize things

034376  260 17 0 00 000000*       CMD1:   GO      .INPNT                  ; print interrupt fielded (if any)
034377  402 00 0 00 064625'               SETZM   PCMD#                   ; clear the command storage
034400  400 15 0 00 000000                SETZ    ERFLG,                  ; clear error flag
034401  201 00 0 00 023420                MOVEI   ^D10000                 ; re-set up print limit just in case
034402  212 00 0 00 030220                MOVNM   PNTENB                  ;   the last test exceeded it

                                  ; Set/clear debug switches

034403  402 00 0 00 064654'       CMD2:   SETZM   UDEBUG                  ; clear no-port user mode debug switch
034404  402 00 0 00 064614'               SETZM   MDEBUG                  ; clear error message debug switch
034405  402 00 0 00 064641'               SETZM   SDEBUG                  ; clear single step debug switch
034406  402 00 0 00 064610'               SETZM   LDEBUG                  ; clear ucode load debug switch
034407  402 00 0 00 064606'               SETZM   IDEBUG                  ; fault isolation debug switch
034410  260 17 0 00 033312*               GO      SWITT                   ; get switches
034411  602 00 0 00 002000                TRNE    DEBUG1                  ; debug switch set?
034412  476 00 0 00 064654'               SETOM   UDEBUG                  ; set no-port user mode debug switch
034413  602 00 0 00 001000                TRNE    DEBUG2                  ; debug switch set?
034414  476 00 0 00 064614'               SETOM   MDEBUG                  ; set error message debug switch
034415  602 00 0 00 000400                TRNE    DEBUG3                  ; debug switch set?
034416  476 00 0 00 064641'               SETOM   SDEBUG                  ; set single step debug switch
034417  602 00 0 00 000200                TRNE    DEBUG4                  ; debug switch set?
034420  476 00 0 00 064610'               SETOM   LDEBUG                  ; set ucode load debug switch
034421  602 00 0 00 000100                TRNE    DEBUG5                  ; debug switch set?
034422  476 00 0 00 064606'               SETOM   IDEBUG                  ; set fault isolation debug switch
034423  402 00 0 00 047034                SETZM   PMODE                   ; set program mode to 'no test'
034424  402 00 0 00 064616'               SETZM   MULFLG                  ; clear 'multiple examine' flag
034425  476 00 0 00 064617'               SETOM   MULINI                  ; set 'initial examine' flag

                                  ; Input command

034426  260 17 0 00 030747*               GO      CLRBUF                  ; clear input buffer
                                  CMD3:   MOVEI   [PCRL                   ; set up $ transfer address
                                                   SETZM   ALTF
                                                   GO      FIOFF
034427  201 00 0 00 055472                         JRST    CMD1]
034430  202 00 0 00 030063                MOVEM   ALTMGO
034431  402 00 0 00 064570'               SETZM   ALTF                    ; clear 'altmode typed' flag
034432  336 00 0 00 064603'               SKIPN   FINPUT                  ; file input?
034433  476 00 0 00 064602'               SETOM   FINECH                  ; no - ensure echo flag set
034434  332 00 0 00 064602'               SKIPE   FINECH                  ; echo flag set?
                                          GO      [PFORCE                 ; no - print all this
                                                   PCRLF
                                                   FMSG    (DEBUG> )
034435  260 17 0 00 055500                         RTN]
034436  332 00 0 00 064603'               SKIPE   FINPUT                  ; file input?
                                          JRST    [GO      FINCMD         ; yes - input sixbit cmd
                                                   JRST    CMD3           ; error or EOF
034437  254 00 0 00 055504                         JRST    CMD4]          ; continue
```

```

034440  037 10 0 00 000003      CMD3A:  TTSIXB                                   ; get operator input
                                        ALTCHK  [GET    XXW#                     ; adjust stack properly
                                                 SKIPN  $TWCNT                   ; did a timeout occur?
                                                 JRST   CMD3                     ; yes - keep looking for input
                                                 MOVE   1,$TTCHR                 ; no - get character typed
                                                 CAIN   1,77                     ; was a question mark typed?
                                                 JRST   CMDQUE                   ; yes - handle
                                                 CAIN   1,40                     ; space (delimiter character) typed?
                                                 JRST   .+1                      ; yes - continue (no error occurred)
                                                 FMSGCD <? Unrecognizable - Type ? for help>
                                                 GO     FIOFF                    ; turn off file input
034441  007 00 0 00 055507                       JRST   CMD1]                    ; error - reask question
                                CMD4:   JUMPE   [MOVE   $TTCHR                   ; anything typed?  if a space, keep
                                                 CAIN   40                       ;   looking for an argument, if not
                                                 JRST   CMD3A                    ;   exit
034442  322 00 0 00 055522                       JRST   CMD3]
034443  202 00 0 00 064625'             MOVEM   PCMD                             ; save command typed
034444  402 00 0 00 033110*             SETZM   ARGFLG                           ; clear 'argument given' flag
034445  200 01 0 00 030231              MOVE    1,$TTCHR                         ; get last character typed
034446  302 01 0 00 000015              CAIE    1,15                             ; CR? (end of line)
034447  476 00 0 00 034444*             SETOM   ARGFLG                           ; set 'argument given' flag

                                ; Decode the command

034450  200 01 0 00 064625'             MOVE    1,PCMD                           ; get sixbit command
034451  201 02 0 00 034776              MOVEI   2,.DBCMD                         ; get address of cmd list
034452  260 17 0 00 034357*             GO      .COMM                            ; handle command decoding
                                         JRST   [FMSGCD <? Unrecognizable - Type ? for help>
                                                 GO     FIOFF
034453  254 00 0 00 055517                       JRST   CMD1]                    ; error return
034454  260 17 1 01 035146              GO      @.DBDIS(1)                       ; dispatch to test
034455  254 00 0 00 034376              JRST    CMD1                             ; return to 'DEBUG>' prompt
034456  254 00 0 00 034371              JRST    CMD0                             ; do port handling again

                                ; Question mark typed

034457  037 01 0 00 000026      CMDQUE: PFORCE                                   ; handle Control-O
034460  037 00 0 00 034462              PNTMSG  CMDQUU                           ; print text
034461  254 00 0 00 034376              JRST    CMD1                             ; look for new command
```

```
2917
2918 034462                           CMDQUU: ASCIZ   /
2919 034462  040 040 040 015 012      --------- Miscellaneous --------------------    -- Switches -
2920                                  HELP    IPRINT  START   EEBUF   DCRAM   LOAD    NOPNT   TRACE
2921                                  EXIT    ICLEAR  STOP    DEBUF   BCRAM   LIST    PNTLPT  INHFLT
2922                                  DDT     IINIT   CONT    ZEBUF   ACRAM   FLOAD   DING    INHMSG
2923                                  SET     SSTEP                   ECRAM   FVERFY  LOOPER  LOOPGM
2924                                  TAKE    RDINIT  STRACE  ELAR    LCRAM   FLIST   ERSTOP  LOOPTS
2925                                  NTAKE   WRINIT  SPRINT  DRAR    ZCRAM   FEXAM   PALERS  RUNALL
2926                                  TRANSL  BPRINT  SCLEAR  ZRAR                    RELIAB  DSPEAR
2927                                  FILLNX  CCWPNT  SSINIT          LCONI   BREAK   TXTINH  UDEBUG
2928                                  FILLPC  LOGPNT          ELOCS   LCONO   RBREAK  OPRSEL  MDEBUG
2929                                          COMPAR  RESET   DLOCS   LDATAI  CBREAK          SDEBUG
2930                                  MARK            CLEAR   ZLOCS   LDATAO  LBREAK  NIPORT  LDEBUG
2931                                  RMARK   CONFIG                  LROUTN          CIPORT  IDEBUG
2932                                  CMARK   CONPNT  E2901   ECSR            CCODE           MMPROC
2933                                  LMARK   SELECT  D2901   DCSR    ENEXT           SWITCH  MCBUS
2934                                                  Z2901   ZCSR    DNEXT   ESTACK  SWPRIN
2935                                  /
```

```
                                        ;#********************************************************************
                                        ;* .DBCMD - List of acceptable commands
                                        ;#********************************************************************

                                        ; Miscellaneous commands

034776  777777  777655                  .DBCMD: -123                            ; ignore
034777  50 45 54 60 00 00                       SIXBIT  /HELP/                  ; print help message
035000  45 70 51 64 00 00                       SIXBIT  /EXIT/                  ; exit DEBUG mode
035001  44 44 64 00 00 00                       SIXBIT  /DDT/                   ; enter DDT

035002  63 45 64 00 00 00                       SIXBIT  /SET/                   ; set/print options
035003  64 41 53 45 00 00                       SIXBIT  /TAKE/                  ; take commands from a file
035004  56 64 41 53 45 00                       SIXBIT  /NTAKE/                 ; take commands from a file
035005  45 56 45 70 64 00                       SIXBIT  /ENEXT/                 ; examine next location
035006  44 56 45 70 64 00                       SIXBIT  /DNEXT/                 ; examine next location
035007  43 57 56 46 51 47                       SIXBIT  /CONFIG/                ; determine/print configuration
035010  43 57 56 60 56 64                       SIXBIT  /CONPNT/                ; print configuration
035011  63 45 54 45 43 64                       SIXBIT  /SELECT/                ; select device to test

                                        ; CSR commands

035012  45 43 63 62 00 00                       SIXBIT  /ECSR/                  ; examine CSR register
035013  44 43 63 62 00 00                       SIXBIT  /DCSR/                  ; deposit CSR register
035014  72 43 63 62 00 00                       SIXBIT  /ZCSR/                  ; zero CSR register

035015  62 45 63 45 64 00                       SIXBIT  /RESET/                 ; issue an EBUS Reset
035016  43 54 45 41 62 00                       SIXBIT  /CLEAR/                 ; issue a 'Port Clear'

                                        ; Start/stop/single step commands

035017  63 64 41 62 64 00                       SIXBIT  /START/                 ; start the port
035020  63 64 57 60 00 00                       SIXBIT  /STOP/                  ; stop the port
035021  43 57 56 64 00 00                       SIXBIT  /CONT/                  ; continue the port
035022  63 63 64 45 60 00                       SIXBIT  /SSTEP/                 ; single step the port
035023  63 64 62 41 43 45                       SIXBIT  /STRACE/                ; single step the port while tracing
035024  63 60 62 51 56 64                       SIXBIT  /SPRINT/                ; print single step history
035025  63 43 54 45 41 62                       SIXBIT  /SCLEAR/                ; clear single step history data
035026  63 63 51 56 51 64                       SIXBIT  /SSINIT/                ; set up ss start addr/data/history

                                        ; EBUF commands

035027  45 45 42 65 46 00                       SIXBIT  /EEBUF/                 ; examine EBUF register
035030  44 45 42 65 46 00                       SIXBIT  /DEBUF/                 ; deposit EBUF register
035031  72 45 42 65 46 00                       SIXBIT  /ZEBUF/                 ; zero EBUF register

                                        ; RAR/LAR commands

035032  45 54 41 62 00 00                       SIXBIT  /ELAR/                  ; examine LAR register
035033  44 62 41 62 00 00                       SIXBIT  /DRAR/                  ; deposit RAR register
035034  72 62 41 62 00 00                       SIXBIT  /ZRAR/                  ; zero RAR register
```

```
                                        ; CRAM commands

        035035  44 43 62 41 55 00              SIXBIT  /DCRAM/                 ; deposit into port CRAM
        035036  42 43 62 41 55 00              SIXBIT  /BCRAM/                 ; deposit into port CRAM by bit field
        035037  41 43 62 41 55 00              SIXBIT  /ACRAM/                 ; alter port CRAM
        035040  45 43 62 41 55 00              SIXBIT  /ECRAM/                 ; examine port CRAM
        035041  54 43 62 41 55 00              SIXBIT  /LCRAM/                 ; list port CRAM
        035042  72 43 62 41 55 00              SIXBIT  /ZCRAM/                 ; zero port CRAM

                                        ; 2901 command

        035043  45 22 31 20 21 00              SIXBIT  /E2901/                 ; examine 2901 registers
        035044  44 22 31 20 21 00              SIXBIT  /D2901/                 ; deposit 2901 registers
        035045  72 22 31 20 21 00              SIXBIT  /Z2901/                 ; zero 2901 registers

                                        ; Local storage commmands

        035046  45 54 57 43 63 00              SIXBIT  /ELOCS/                 ; examine local storage
        035047  44 54 57 43 63 00              SIXBIT  /DLOCS/                 ; deposit local storage
        035050  72 54 57 43 63 00              SIXBIT  /ZLOCS/                 ; zero local storage

                                        ; Scope looping commands

        035051  54 43 57 56 51 00              SIXBIT  /LCONI/                 ; loop on CONI
        035052  54 43 57 56 57 00              SIXBIT  /LCONO/                 ; loop on CONO
        035053  54 44 41 64 41 51              SIXBIT  /LDATAI/                ; loop on DATAI
        035054  54 44 41 64 41 57              SIXBIT  /LDATAO/                ; loop on DATAO
        035055  54 62 57 65 64 56              SIXBIT  /LROUTN/                ; loop on routine

                                        ; Mark bit commands

        035056  55 41 62 53 00 00              SIXBIT  /MARK/                  ; set mark bit in locations
        035057  62 55 41 62 53 00              SIXBIT  /RMARK/                 ; remove mark bit in locations
        035060  43 55 41 62 53 00              SIXBIT  /CMARK/                 ; clear mark bit in locations
        035061  54 55 41 62 53 00              SIXBIT  /LMARK/                 ; list locations with mark bit set

                                        ; Breakpoint commands

        035062  42 62 45 41 53 00              SIXBIT  /BREAK/                 ; set breakpoint
        035063  62 42 62 45 41 53              SIXBIT  /RBREAK/                ; remove breakpoint
        035064  43 42 62 45 41 53              SIXBIT  /CBREAK/                ; clear all breakpoints
        035065  54 42 62 45 41 53              SIXBIT  /LBREAK/                ; list breakpoints

                                        ; Microcode file commands

        035066  54 57 41 44 00 00              SIXBIT  /LOAD/                  ; load microcode from test into CRAM
        035067  54 51 63 64 00 00              SIXBIT  /LIST/                  ; list microcode from test
        035070  46 54 57 41 44 00              SIXBIT  /FLOAD/                 ; load microcode from file into CRAM
        035071  46 66 45 62 46 71              SIXBIT  /FVERFY/                ; verify microcode loaded
        035072  46 54 51 63 64 00              SIXBIT  /FLIST/                 ; list microcode in file
        035073  46 45 70 41 55 00              SIXBIT  /FEXAM/                 ; examine microcode in file
```

```


                                       ; Miscellaneous commands

035074  64 62 41 56 63 54                      SIXBIT  /TRANSL/            ; translate CSR bits
035075  43 43 57 44 45 00                      SIXBIT  /CCODE/             ; print condition codes
035076  45 63 64 41 43 53                      SIXBIT  /ESTACK/            ; examine 2910 stack contents
035077  46 51 54 54 56 70                      SIXBIT  /FILLNX/            ; fill CRAM with special data
035100  46 51 54 54 60 43                      SIXBIT  /FILLPC/            ; fill CRAM with special data
035101  51 60 62 51 56 64                      SIXBIT  /IPRINT/            ; print interrupt history
035102  51 43 54 45 41 62                      SIXBIT  /ICLEAR/            ; clear interrupt history
035103  51 51 56 51 64 00                      SIXBIT  /IINIT/             ; initialize interrupt system

                                       ; Data transfer routines

035104  62 44 51 56 51 64                      SIXBIT  /RDINIT/            ; initialize e/g for a read
035105  67 62 51 56 51 64                      SIXBIT  /WRINIT/            ; initialize e/g for a write
035106  42 60 62 51 56 64                      SIXBIT  /BPRINT/            ; print buffer contents
035107  43 43 67 60 56 64                      SIXBIT  /CCWPNT/            ; print CCW list
035110  54 57 47 60 56 64                      SIXBIT  /LOGPNT/            ; print logout data
035111  43 57 55 60 41 62                      SIXBIT  /COMPAR/            ; compare data buffer

                                       ; Switches

035112  63 67 51 64 43 50                      SIXBIT  /SWITCH/            ; input switches
035113  63 67 60 62 51 56                      SIXBIT  /SWPRIN/            ; print switch selections
035114  56 57 60 56 64 00                      SIXBIT  /NOPNT/             ; don't print anything
035115  60 56 64 54 60 64                      SIXBIT  /PNTLPT/            ; print on printer
035116  44 51 56 47 00 00                      SIXBIT  /DING/              ; ding on error
035117  54 57 57 60 45 62                      SIXBIT  /LOOPER/            ; loop on error
035120  45 62 63 64 57 60                      SIXBIT  /ERSTOP/            ; halt on error
035121  60 41 54 45 62 63                      SIXBIT  /PALERS/            ; print all errors
035122  62 45 54 51 41 42                      SIXBIT  /RELIAB/            ; reliability testing
035123  64 70 64 51 56 50                      SIXBIT  /TXTINH/            ; shorten printing
035124  57 60 62 63 45 54                      SIXBIT  /OPRSEL/            ; operator select
035125  64 62 41 43 45 00                      SIXBIT  /TRACE/             ; trace pgm flow
035126  51 56 50 46 54 64                      SIXBIT  /INHFLT/            ; inhibit fault isolation
035127  51 56 50 55 63 47                      SIXBIT  /INHMSG/            ; inhibit error messages
035130  54 57 57 60 47 55                      SIXBIT  /LOOPGM/            ; loop on program
035131  54 57 57 60 64 63                      SIXBIT  /LOOPTS/            ; loop on test
035132  62 65 56 41 54 54                      SIXBIT  /RUNALL/            ; run all test segments
035133  44 63 60 45 41 62                      SIXBIT  /DSPEAR/            ; disable SPEAR reporting
035134  65 44 45 42 65 47                      SIXBIT  /UDEBUG/            ; no-port user mode debug switch
035135  55 44 45 42 65 47                      SIXBIT  /MDEBUG/            ; error message debug switch
035136  63 44 45 42 65 47                      SIXBIT  /SDEBUG/            ; single step debug switch
035137  54 44 45 42 65 47                      SIXBIT  /LDEBUG/            ; ucode load debug switch
035140  51 44 45 42 65 47                      SIXBIT  /IDEBUG/            ; fault isolation debug switch
035141  56 51 60 57 62 64                      SIXBIT  /NIPORT/            ; NI port selected
035142  43 51 60 57 62 64                      SIXBIT  /CIPORT/            ; CI port selected
035143  55 55 60 62 57 43                      SIXBIT  /MMPROC/            ; missing Mproc module
035144  55 43 42 65 63 00                      SIXBIT  /MCBUS/             ; missing Cbus module
035145  000 00 0 00 000000                     Z                           ; list terminator
```

```


                                   ;#********************************************************************
                                   ;* .DBDIS - List of dispatch addresses
                                   ;#********************************************************************

035146  000000  052526             .DBDIS: [RTN]                                   ; ignore
035147  000000  035315                     DHELP                                   ; HELP - print help message
                                           [MOVE     1,[FMSGCD (EXIT <CR>)]        ; EXIT - exit DEBUG mode
                                            GO       CHKARG
                                            RTN
                                            GET      XXX#
                                            RTN]
035150  000000  055526                     TSTDDT                                  ; DDT - enter DDT
035151  000000  032272
                                           DSET                                    ; SET - set/clear/print options
035152  000000  035403                     TSTTAK                                  ; TAKE - take commands from file
035153  000000  032715                     TSTTAN                                  ; NTAKE - take commands from a file
035154  000000  032713                     DENEXT                                  ; ENEXT - examine next location
035155  000000  041455                     DDNEXT                                  ; DNEXT - deposit next location
035156  000000  041532                     [GO       CONFIG                        ; CONFIG - determine configuration
                                            GO       TSTSEL
                                            SETZM    MBCN
                                            AOS      (P)
                                            RTN]
035157  000000  055533                     [MOVE     1,[FMSGCD (CONPNT <CR>)];     CONPNT - print configuration
                                            GO       CHKARG
                                            RTN
                                            GO       CONPNT
                                            RTN]
035160  000000  052623                     [GO       TSTSEL                        ; SELECT - select device to test
                                            SETZM    MBCN
                                            AOS      (P)
                                            RTN]
035161  000000  055534                     DECSR                                   ; ECSR - examine CSR register
035162  000000  035771                     DDCSR                                   ; DCSR - deposit CSR register
035163  000000  036001                     DZCSR                                   ; ZCSR - zero CSR register
035164  000000  036010
                                           DRESET                                  ; RESET - issue an EBUS Reset
035165  000000  036106                     DCLEAR                                  ; CLEAR - issue a 'Port Clear'
035166  000000  036113
                                           DSTART                                  ; START - start the port
035167  000000  036133                     DSTOP                                   ; STOP - stop the port
035170  000000  036160                     DCONT                                   ; CONT - continue the port
035171  000000  037551                     DSSTEP                                  ; SSTEP - single step the port
035172  000000  036203                     DTRACE                                  ; STRACE - sstep the port while tracing
035173  000000  036200                     DSSPNT                                  ; SPRINT - print single step history
035174  000000  036252                     DSSCLR                                  ; SCLEAR - clear single step history
035175  000000  036274                     DSINIT                                  ; SSINIT - set up ss addr/data/history
035176  000000  036120
                                           DEEBUF                                  ; EEBUF - examine EBUF register
035177  000000  037566                     DDEBUF                                  ; DEBUF - deposit EBUF register
035200  000000  037607                     DZEBUF                                  ; ZEBUF - zero EBUF register
035201  000000  037602
```

```
035202  000000  037621                  DELAR                           ; ELAR - examine LAR register
035203  000000  037641                  DDRAR                           ; DRAR - deposit RAR register
035204  000000  037653                  DZRAR                           ; ZRAR - zero RAR register

035205  000000  037666                  DDCRAM                          ; DCRAM - deposit CRAM
035206  000000  037671                  DBCRAM                          ; BCRAM - deposit CRAM by bit field
035207  000000  037674                  DACRAM                          ; ACRAM - alter port CRAM
035210  000000  037663                  DECRAM                          ; ECRAM - examine port CRAM
035211  000000  037702                  DLCRAM                          ; LCRAM - list port CRAM
035212  000000  037677                  DZCRAM                          ; ZCRAM - zero port CRAM

035213  000000  040451                  DEALU                           ; E2901 - examine 2901 registers
035214  000000  040557                  DDALU                           ; D2901 - deposit 2901 registers
035215  000000  040640                  DZALU                           ; Z2901 - zero 2901 registers

035216  000000  040755                  DELOCS                          ; ELOCS - examine local storage
035217  000000  041060                  DDLOCS                          ; DLOCS - deposit local storage
035220  000000  041134                  DZLOCS                          ; ZLOCS - zero local storage

035221  000000  036016                  DCONI                           ; LCONI - loop on CONI
035222  000000  036026                  DCONO                           ; LCONO - loop on CONO
035223  000000  036037                  DDATI                           ; LDATAI - loop on DATAI
035224  000000  036046                  DDATO                           ; LDATAO - loop on DATAO
035225  000000  036057                  DROUTN                          ; LROUTN - loop on routine

035226  000000  041250                  DMARK                           ; MARK - set mark bits
035227  000000  041277                  DRMARK                          ; RMARK - remove mark bits
035230  000000  041326                  DCMARK                          ; CMARK - clear mark bits
035231  000000  041377                  DLMARK                          ; LMARK - list mark bits

035232  000000  041634                  DBREAK                          ; BREAK - set breakpoint
035233  000000  041667                  DRBRK                           ; RBREAK - remove breakpoint
035234  000000  041712                  DCBRK                           ; CBREAK - clear all breakpoints
035235  000000  041756                  DLBRK                           ; LBREAK - list breakpoints

035236  000000  042037                  DLOAD                           ; LOAD - load test microcode in CRAM
035237  000000  042062                  DLIST                           ; LIST - list test microcode
035240  000000  042077                  DFLOAD                          ; FLOAD - load ucode from file to CRAM
035241  000000  042135                  DFVERF                          ; FVERFY - verify microcode loaded
035242  000000  042227                  DFLIST                          ; FLIST - list microcode in file
035243  000000  042256                  DFEXAM                          ; FEXAM - examine microcode in file

035244  000000  042304                  DTRANS                          ; TRANSL - translate CSR bits
035245  000000  042314                  DCCODE                          ; CCODE - print condition codes
035246  000000  041574                  DESTAK                          ; ESTACK - examine 2910 stack
035247  000000  042443                  DFILLN                          ; FILLNX - fill CRAM with special data
035250  000000  042466                  DFILLP                          ; FILLPC - fill CRAM with special data
035251  000000  042635                  DIPRIN                          ; IPRINT - print interrupt history
                                        [MOVE   1,[FMSGCD (ICLEAR <CR>)]; ICLEAR - clear interrupt history
                                         GO     CHKARG
                                         RTN
```

```
                                           SETZM   INTNUM
                                   MOVE    [INTNUM,,INTNUM+1]
                                   BLT     INTEND
                                   SETOM   INTNUM
035252  000000  055546             RTN]
035253  000000  042711             DIINIT                          ; IINIT - initialize interrupt system

035254  000000  042507             DRDINI                          ; RDINIT - initialize for a read
035255  000000  042532             DWRINI                          ; WRINIT - initialize for a write
035256  000000  042537             DBPNT                           ; BPRINT - print buffer data
035257  000000  042572             DCWPNT                          ; CCWPNT - print CCW list
035260  000000  042600             DLGPNT                          ; LOGPNT - print logout data
035261  000000  042610             DCOMPA                          ; COMPAR - compare data buffer

                                   [MOVE   1,[FMSGCD (SWITCH <CR>)]; SWITCH - input switches
                                    GO     CHKARG
                                    RTN
                                    GO     .ISWT
035262  000000  055556              RTN]
                                   [MOVE   1,[FMSGCD (SWPRIN <CR>)]; SWPRIN - print switches
                                    GO     CHKARG
                                    RTN
                                    GO     .SWCHP
035263  000000  055563              RTN]

                000116        SWSTP==.-.DBDIS

035264  000000  042624             DSWIT                           ; NOPNT
035265  000000  042624             DSWIT                           ; PNTLPT
035266  000000  042624             DSWIT                           ; DING
035267  000000  042624             DSWIT                           ; LOOPER
035270  000000  042624             DSWIT                           ; ERSTOP
035271  000000  042624             DSWIT                           ; PALERS
035272  000000  042624             DSWIT                           ; RELIAB
035273  000000  042624             DSWIT                           ; TXTINH
035274  000000  042624             DSWIT                           ; OPRSEL
035275  000000  042624             DSWIT                           ; TRACE
035276  000000  042624             DSWIT                           ; INHFLT
035277  000000  042624             DSWIT                           ; INHMSG
035300  000000  042624             DSWIT                           ; LOOPGM
035301  000000  042624             DSWIT                           ; LOOPTS
035302  000000  042624             DSWIT                           ; RUNALL
035303  000000  042624             DSWIT                           ; DSPEAR
035304  000000  042624             DSWIT                           ; UDEBUG
035305  000000  042624             DSWIT                           ; MDEBUG
035306  000000  042624             DSWIT                           ; SDEBUG
035307  000000  042624             DSWIT                           ; LDEBUG
035310  000000  042624             DSWIT                           ; IDEBUG
035311  000000  042624             DSWIT                           ; NIPORT
035312  000000  042624             DSWIT                           ; CIPORT
035313  000000  042624             DSWIT                           ; MMPROC
035314  000000  042624             DSWIT                           ; MCBUS
```

```

                                      ;***********************************************************************
                                      ;* DHELP - Print DEBUG> help message
                                      ;***********************************************************************


                                      DHELP:  MOVN    1,[FMSGD <
                                      HELP ALL       RESET  LAR    SCOPE  DTXFR
                                              MISC   START  CRAM   MARK   SWITCH
                                              INTRPT CSR    2901   BREAK
                                              CONFIG EBUF   LOCS   UCODE>]
035315  210 01 0 00 055626                    GO      .SARG                     ; get SIXBIT argument
035316  260 17 0 00 032725*                    RTN                              ; error/altmode/question - exit
035317  263 17 0 00 000000                     GO     [MOVE   [SIXBIT /ALL/]    ; no argument - assume all
                                                        MOVEM  ARGUM
                                                        RTN]
035320  260 17 0 00 054543
035321  200 01 0 00 033062*                   MOVE    1,ARGUM                   ; get sixbit command
035322  201 02 0 00 035335                    MOVEI   2,.HLCMD                  ; get address of cmd list
035323  260 17 0 00 034452*                   GO      .COMM                     ; handle command decoding
                                               JRST   [FMSGCD <? Unrecognizable - Type ? for help>
                                                        GO      FIOFF
                                                        RTN]                    ; error return
035324  254 00 0 00 054410                    PFORCE                            ; handle Control-O
035325  037 01 0 00 000026                    MOVE    2,1                       ; get AC1
035326  200 02 0 00 000001            DHELP0: SKIPN   3,DHELP1(1)               ; get address of TMSG
035327  336 03 0 01 035360                    RTN                               ; all done - exit
035330  263 17 0 00 000000                    XCT     3                         ; print it
035331  256 00 0 00 000003                    AOS     1                         ; point to next address
035332  350 00 0 00 000001                    JUMPE   2,DHELP0                  ; loop till done
035333  322 02 0 00 035327                    RTN                               ; exit
035334  263 17 0 00 000000

035335  41 54 54 00 00 00             .HLCMD: SIXBIT  /ALL/                     ; print all help info
035336  55 51 63 43 00 00                     SIXBIT  /MISC/                    ; print miscellaneous commands
035337  51 56 64 62 60 64                     SIXBIT  /INTRPT/                  ; print interrupt commands
035340  43 57 56 46 51 47                     SIXBIT  /CONFIG/                  ; print configuration commands
035341  62 45 63 45 64 00                     SIXBIT  /RESET/                   ; print reset commands
035342  63 64 41 62 64 00                     SIXBIT  /START/                   ; print start... commands
035343  43 63 62 00 00 00                     SIXBIT  /CSR/                     ; print CSR commands
035344  45 42 65 46 00 00                     SIXBIT  /EBUF/                    ; print EBUF commands
035345  54 41 62 00 00 00                     SIXBIT  /LAR/                     ; print LAR commands
035346  43 62 41 55 00 00                     SIXBIT  /CRAM/                    ; print CRAM commands
035347  22 31 20 21 00 00                     SIXBIT  /2901/                    ; print 2901 commands
035350  54 57 43 63 00 00                     SIXBIT  /LOCS/                    ; print local storage commands
035351  63 43 57 60 45 00                     SIXBIT  /SCOPE/                   ; print scope looping commands
035352  55 41 62 53 00 00                     SIXBIT  /MARK/                    ; print mark commands
035353  42 62 45 41 53 00                     SIXBIT  /BREAK/                   ; print breakpoint commands
035354  65 43 57 44 45 00                     SIXBIT  /UCODE/                   ; print microcode commands
035355  44 64 70 46 62 00                     SIXBIT  /DTXFR/                   ; print data transfer commands
035356  63 67 51 64 43 50                     SIXBIT  /SWITCH/                  ; print switch commands
035357  000000  000000                        0                                 ; end of list
```

```

035360  037 00 0 00 055627      DHELP1: TMSGCD  <DEBUG Commands:>

                                                TMSGD   <
                                HELP arg    Print help
                                EXIT        Exit DEBUG mode
                                DDT         Enter DDT
                                SET opt arg    Set/clear/print options
                                TAKE file ext  Take commands from file
                                NTAKE file ext Take commands from file (unechoed)
                                TRANSL dat  Translate CSR data
                                CCODE       Print condition codes
                                ESTACK      Examine 2910 stack contents
                                FILLNX dat  Fill CRAM with 'J=.+1,CTL=dat'
                                FILLPC dat  Fill CRAM with 'J=.,CTL=dat'
                                ENEXT       Examine next
035361  037 00 0 00 055633      DNEXT arg   Deposit next>

                                                TMSGD   <
                                IPRINT      Print accumulated interrupt activity
                                ICLEAR      Clear accumulated interrupt activity
035362  037 00 0 00 055767      IINIT       Init interrupt system>

                                                TMSGD   <
                                CONFIG      Determine/print configuration
                                CONPNT      Print configuration
035363  037 00 0 00 056023      SELECT      Select device to test>

                                                TMSGD   <
                                RESET       Issue EBUS Reset
035364  037 00 0 00 056052      CLEAR       Issue a 'Port Clear'>

                                                TMSGD   <
                                START adr   Start the port microsequencer
                                STOP        Stop the port microsequencer
                                CONT        Continue the port microsequencer
                                SSTEP cnt   Single step the port microsequencer
                                STRACE cnt  Single step and trace execution
                                SPRINT cnt  Print single step history data
                                SCLEAR      Clear single step history data
035365  037 00 0 00 056070      SSINIT      Set up ss addr/data/history>

                                                TMSGD   <
                                ECSR        Examine CSR Register
                                DCSR dat    Deposit CSR Register
035366  037 00 0 00 056200      ZCSR        Zero CSR Register>

                                                TMSGD   <
                                EEBUF       Examine EBUF register
                                DEBUF dat   Deposit EBUF register
035367  037 00 0 00 056225      ZEBUF       Zero EBUF register>

```

```
                                                  TMSGD   <
                                          ELAR         Examine LAR register
                                          DRAR dat     Deposit RAR register
035370  037 00 0 00 056252                ZRAR         Zero RAR register>

                                                  TMSGD   <
                                          DCRAM adr,adr  Deposit into CRAM
                                          BCRAM adr,adr  Deposit into CRAM by fields
                                          ACRAM adr,adr  Alter CRAM
                                          ECRAM adr,adr  Examine CRAM
                                          LCRAM adr,adr  List CRAM
035371  037 00 0 00 056277                ZCRAM adr,adr  Zero CRAM>

                                                  TMSGD   <
                                          E2901 reg,reg  Examine 2901 registers
                                          D2901 reg,dat  Deposit 2901 register
035372  037 00 0 00 056345                Z2901 reg,reg  Zero 2901 registers>

                                                  TMSGD   <
                                          ELOCS adr,adr  Examine local storage
                                          DLOCS adr,dat  Deposit local storage
035373  037 00 0 00 056375                ZLOCS adr,adr  Zero local storage>

                                                  TMSGD   <
                                          LCONI        Loop on CONI
                                          LCONO dat    Loop on CONO
                                          LDATAI       Loop on DATAI
                                          LDATAO dat   Loop on DATAO
035374  037 00 0 00 056424                LROUTN dat   Loop on user supplied routine at DROUTN:>

                                                  TMSGD   <
                                          MARK adr,adr   Set mark bit
                                          RMARK adr,adr  Remove mark bit
                                          CMARK adr,adr  Clear all mark bits
035375  037 00 0 00 056465                LMARK adr,adr  List mark bits>

                                                  TMSGD   <
                                          BREAK adr,adr  Set breakpoint
                                          RBREAK adr,adr Remove breakpoint
                                          CBREAK adr,adr Clear all breakpoints
035376  037 00 0 00 056520                LBREAK adr,adr List breakpoints>

                                                  TMSGD   <
                                          LOAD tst     Load microcode from Test 'tst' into CRAM
                                          LIST tst     List microcode from Test 'tst'
                                          FLOAD        Load microcode from file into CRAM
                                          FVERFY       Verify microcode loaded from file
                                          FLIST        List microcode in file
035377  037 00 0 00 056554                FEXAM        Examine microcode in file>
```

```
                                     TMSGD   <
                                RDINIT    Initialize e/g for a read
                                WRINIT    Initialize e/g for a write
                                BPRINT n  Print buffer contents (n locations)
                                CCWPNT    Print CCW list
                                LOGPNT    Print logout data
035400  037 00 0 00 056643      COMPAR    Compare data buffer>

                                     TMSGD   <
                                SWITCH  Enter switch information
                                SWPRIN  Print current switch settings

                                (To complement switch setting, type switch name).

                                NOPNT   LOOPER  RELIAB   TRACE   LOOPGM  DSPEAR  SDEBUG  NIPORT  MMPROC
                                PNTLPT  ERSTOP  TXTINH   INHFLT  LOOPTS  UDEBUG  LDEBUG  CIPORT  MCBUS
035401  037 00 0 00 056716      DING    PALERS  OPRSEL   INHMSG  RUNALL  MDEBUG  IDEBUG>
035402  000000  000000                  0
```

```

                                        ;#*************************************************************
                                        ;* DSET - Set/clear/print options
                                        ;#*************************************************************

                                        DSET:   MOVE    1,[FMSGD <
                                        SET ADDR    EBUF    PATLIS   WHAT
                                            CSR     LENGTH  FNAME    HELP
                                            PARITY  PAT>]
035403  200 01 0 00 057043                      GO      .SARG                   ; get SIXBIT argument
035404  260 17 0 00 035316*                      RTN                            ; error/altmode/question - exit
035405  263 17 0 00 000000                       JRST   [FMSGCD <? Missing argument>
                                                         GO     FIOFF
                                                .        RTN]
035406  254 00 0 00 055164
035407  200 01 0 00 035321*             MOVE    1,ARGUM                 ; get sixbit command
035410  201 02 0 00 035415              MOVEI   2,.STCMD                ; get address of cmd list
035411  260 17 0 00 035323*             GO      .COMM                   ; handle command decoding
                                         JRST   [FMSGCD <? Unrecogniz ble - Type ? for help>
                                                 GO     FIOFF
035412  254 00 0 00 054410                       RTN]                   ; error return
035413  260 17 1 01 035431              GO      @.STDIS(1)              ; dispatch to test
035414  263 17 0 00 000000              RTN                             ; return to 'DEBUG>' prompt

035415  41 44 44 62 00 00       .STCMD: SIXBIT  /ADDR/                  ; set start address
035416  43 63 62 00 00 00               SIXBIT  /CSR/                   ; set start data for CSR
035417  60 41 62 51 64 71               SIXBIT  /PARITY/                ; set/clr auto parity generation
035420  45 42 65 46 00 00               SIXBIT  /EBUF/                  ; set/clr preserve EBUF
035421  54 45 56 47 64 50               SIXBIT  /LENGTH/                ; set length of data transfer
035422  60 41 64 00 00 00               SIXBIT  /PAT/                   ; set data pattern
035423  46 56 41 55 45 00               SIXBIT  /FNAME/                 ; set file name
035424  60 41 64 54 51 63               SIXBIT  /PATLIS/                ; list data patterns
035425  67 50 41 64 00 00               SIXBIT  /WHAT/                  ; print options
035426  50 45 54 60 00 00               SIXBIT  /HELP/                  ; set help message
035427  46 41 65 54 64 00               SIXBIT  /FAULT/                 ; set failing test
035430  000 00 0 00 000000              Z

035431  000000  035444          .STDIS: DSETAD                          ; ADDR - set start address
035432  000000  035465                  DSETCS                          ; CSR - set start CSR data
035433  000000  035502                  DSETPR                          ; PARITY - set/clr auto par generation
035434  000000  035522                  DSETEB                          ; EBUF - set/clr preserve EBUF flag
035435  000000  035542                  DSETLN                          ; LENGTH - set buffer length
035436  000000  035562                  DSETDA                          ; PAT - set data pattern
035437  000000  035622                  DSETFN                          ; FNAME - set default ucode file name
035440  000000  035716                  DSETPL                          ; PATLIS - list data patterns
035441  000000  035652                  DSETWH                          ; WHAT - print options
035442  000000  035753                  DSETHE                          ; HELP - set help message
035443  000000  035666                  DSETFA                          ; FAULT - set failing test
```

```
                                           ;#**********************************************************************
                                           ;* DSETAD - Set start address for START/SSTEP/CONT commands
                                           ;#**********************************************************************

035444  210 01 0 00 057056         DSETAD: MOVN    1,[FMSGCD (SET ADDR addr (0-7777) <CR> or SET ADDR <CR>)]
035445  260 17 0 00 000000*                GO      .OARG                           ; get argument
035446  263 17 0 00 000000                  RTN                                    ; error/altmode/question - exit
035447  254 00 0 00 035457                  JRST   DSETA1                          ; no argument given
035450  200 00 0 00 035407*                MOVE    ARGUM                           ; get argument
035451  202 00 0 00 035762                 MOVEM   DSADDR                          ; save it
035452  301 00 0 00 000000                 CAIL    0                               ; out of range?
035453  303 00 0 00 007777                 CAILE   7777
                                           JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                    GO      FIOFF
035454  254 00 0 00 057070                          RTN]
035455  202 00 0 00 035762                 MOVEM   DSADDR                          ; no - save it
035456  263 17 0 00 000000                 RTN                                     ; exit

035457  037 00 0 00 030242         DSETA1: PCRL
035460  037 00 0 00 057073         DSETA2: TMSG    <Start address ............. >
035461  200 00 0 00 035762                 MOVE    DSADDR
035462  037 16 0 00 000003                 PNTOCS
035463  037 00 0 00 030242                 PCRL
035464  263 17 0 00 000000                 RTN                                     ; exit



                                           ;#**********************************************************************
                                           ;* DSETCS - Set start CSR data for START/CONT commands
                                           ;#**********************************************************************

035465  210 01 0 00 057111         DSETCS: MOVN    1,[FMSGCD (SET CSR xxxxxx <CR> or SET CSR <CR>)]
035466  260 17 0 00 035445*                GO      .OARG                           ; get argument
035467  263 17 0 00 000000                  RTN                                    ; error/altmode/question - exit
035470  254 00 0 00 035474                  JRST   DSETC1                          ; no argument given
035471  200 00 0 00 035450*                MOVE    ARGUM                           ; get argument
035472  202 00 0 00 035763                 MOVEM   DSDATA                          ; save it
035473  263 17 0 00 000000                 RTN                                     ; exit

035474  037 00 0 00 030242         DSETC1: PCRL
035475  037 00 0 00 057112         DSETC2: TMSG    <Start CSR data ............ >
035476  200 00 0 00 035763                 MOVE    DSDATA
035477  037 16 0 00 000003                 PNTOCS
035500  037 00 0 00 030242                 PCRL
035501  263 17 0 00 000000                 RTN                                     ; exit
```

```

                                     ;#*********************************************************************
                                     ;* DSETPR - Set/clear automatic parity generation for load cram command
                                     ;#*********************************************************************

035502  210 01 0 00 057130           DSETPR: MOVN    1,[FMSGCD (SET PAR NO or YES or SET PAR <CR>)]
035503  260 17 0 00 035404*                  GO      .SARG                       ; get argument
035504  263 17 0 00 000000                    RTN                                ; error/altmode/question - exit
035505  254 00 0 00 035513                    JRST   DSETP1                      ; no argument given
035506  260 17 0 00 000000*                  GO      DECYN                       ; decode argument Y or N
035507  263 17 0 00 000000                    RTN                                ; error - unrecognized
                                             JRST    [SETZM  PARFLG              ; no - clear the flag
                                                      RTN]
035510  254 00 0 00 057131
035511  476 00 0 00 035764                   SETOM   PARFLG                      ; yes - set it
035512  263 17 0 00 000000                   RTN                                 ; exit

035513  037 00 0 00 030242           DSETP1: PCRL
035514  037 00 0 00 057133           DSETP2: TMSG    <Auto parity generation .... >
035515  332 00 0 00 035764                   SKIPE   PARFLG                      ; set?
035516  037 00 0 00 057141                   TMSGD   <YES>
035517  336 00 0 00 035764                   SKIPN   PARFLG                      ; set?
035520  037 00 0 00 057143                   TMSGD   <NO>
035521  263 17 0 00 000000                   RTN                                 ; return


                                     ;#*********************************************************************
                                     ;* DSETEB - Set/clear preserve EBUF for single stepping
                                     ;#*********************************************************************

035522  210 01 0 00 057154           DSETEB: MOVN    1,[FMSGCD (SET EBUF NO or YES or SET EBUF <CR>)]
035523  260 17 0 00 035503*                  GO      .SARG                       ; get argument
035524  263 17 0 00 000000                    RTN                                ; error/altmode/question - exit
035525  254 00 0 00 035533                    JRST   DSETE1                      ; no argument given
035526  260 17 0 00 035506*                  GO      DECYN                       ; decode argument Y or N
035527  263 17 0 00 000000                    RTN                                ; error - unrecognized
                                             JRST    [SETZM  PRSFLG              ; no - clear the flag
                                                      RTN]
035530  254 00 0 00 057155
035531  476 00 0 00 035765                   SETOM   PRSFLG                      ; yes - set it
035532  263 17 0 00 000000                   RTN                                 ; exit

035533  037 00 0 00 030242           DSETE1: PCRL
035534  037 00 0 00 057157           DSETE2: TMSG    <EBUF preserved in sstep ... >
035535  332 00 0 00 035765                   SKIPE   PRSFLG                      ; set?
035536  037 00 0 00 057141                   TMSGD   <YES>
035537  336 00 0 00 035765                   SKIPN   PRSFLG                      ; set?
035540  037 00 0 00 057143                   TMSGD   <NO>
035541  263 17 0 00 000000                   RTN                                 ; return
```

```
                                         ;#*********************************************************************
                                         ;* DSETLN - Set length of data transfer
                                         ;#*********************************************************************

035542  210 01 0 00 057177               DSETLN: MOVN    1,[FMSGCD (SET LEN length (1-102400.) or SET LEN <CR>)]
035543  260 17 0 00 033057*                      GO      .DARG                   ; get argument
035544  263 17 0 00 000000                        RTN                            ; error/altmode/question - exit
035545  254 00 0 00 035554                         JRST  DSETL1                  ; no argument given
035546  200 00 0 00 035471*                      MOVE    ARGUM                   ; get argument
035547  303 00 0 00 000000                       CAILE   0                       ; argument in range 1-102400?
035550  303 00 0 00 310000                       CAILE   ^D102400
                                                 JRST    [FMSGCD <? Out of range 1-102400. words>
                                                          GO      FIOFF
035551  254 00 0 00 057207                                RTN]
035552  202 00 0 00 035766                       MOVEM   DSWORD                  ; save it
035553  263 17 0 00 000000                       RTN                             ; exit

035554  037 00 0 00 030242               DSETL1: PCRL
035555  037 00 0 00 057212               DSETL2: TMSG    <Buffer length (words) ..... >
035556  200 00 0 00 035766                       MOVE    DSWORD
035557  037 15 0 00 000000                       PNTDEC
035560  037 00 0 00 055243                       TMSGD   <.>
035561  263 17 0 00 000000                       RTN                             ; exit


                                         ;#*********************************************************************
                                         ;* DSETDA - Set data pattern to insert in buffer
                                         ;#*********************************************************************

035562  200 01 0 00 057235               DSETDA: MOVE    1,[FMSGCD (SET PAT number (1-126 octal)(126=opr sel) or SET PAT <CR>)]
035563  260 17 0 00 035466*                      GO      .DARG                   ; get argument
035564  263 17 0 00 000000                        RTN                            ; error/altmode/question - exit
035565  254 00 0 00 035607                         JRST  DSETD1                  ; no argument given
035566  200 00 0 00 035546*                      MOVE    ARGUM                   ; get argument
035567  301 00 0 00 000001                       CAIL    1                       ; in range?
035570  303 00 0 00 000126                       CAILE   126
                                                 JRST    [FMSGCD <? Out of range 1-126 octal>
                                                          GO      FIOFF
035571  254 00 0 00 057245                                RTN]
035572  202 00 0 00 035767                       MOVEM   DSPAT                   ; save it
035573  302 00 0 00 000126                       CAIE    126                     ; operator selected pattern?
035574  254 00 0 00 035603                       JRST    DSETDX                  ; no - exit
035575  200 01 0 00 057256                       MOVE    1,[FMSGCD <SET PAT 126,data (CR)>]
035576  260 17 0 00 035563*                      GO      .OARG                   ; get argument
035577  263 17 0 00 000000                        RTN                            ; error/altmode/question - exit
                                                  JRST   [FMSGCD <? Missing argument>
                                                          GO      FIOFF
035600  254 00 0 00 055164                                RTN]
```

```
035601  200 00 0 00 035566*             MOVE    ARGUM                   ; get argument
035602  202 00 0 00 035770              MOVEM   DSTPAT                  ; save it
035603  260 17 0 00 000000*     DSETDX: GO      LASARG                  ; check for last argument
035604  263 17 0 00 000000              RTN                             ; exit
035605  037 01 0 00 057250              FMSGCD  <SET PAT 126,data (CR)>
035606  263 17 0 00 000000              RTN                             ; exit

035607  037 00 0 00 030242      DSETD1: PCRL
035610  037 00 0 00 057257      DSETD2: TMSG    <Data pattern (#) .......... >
035611  200 01 0 00 035767              MOVE    1,DSPAT                 ; get pattern
035612  256 00 0 01 000000*             XCT     PATPNT(1)               ;  ...
035613  302 01 0 00 000126              CAIE    1,126                   ; operator selected pattern?
035614  254 00 0 00 035620              JRST    DSETD3                  ; no - continue
035615  037 00 0 00 054243              TMSG    < - >                   ; yes - print it out
035616  200 00 0 00 035770              MOVE    DSTPAT
035617  037 13 0 00 000000              PNTHW
035620  037 00 0 00 030242      DSETD3: PCRL
035621  263 17 0 00 000000              RTN                             ; exit



                                ;#*********************************************************************
                                ;* DSETFN - Set default microcode file name
                                ;#*********************************************************************

035622  200 01 0 00 057277      DSETFN: MOVE    1,[FMSGCD (SET FNAME file ext <CR> or SET FNAME <CR>)]
035623  260 17 0 00 035523*             GO      .SARG                   ; get argument (file name)
035624  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
035625  254 00 0 00 035641               JRST   DSETF1                  ; no argument - print it out
035626  200 00 0 00 035601*             MOVE    ARGUM                   ; get argument
035627  202 00 0 00 064645'             MOVEM   TMP1                    ; save it
035630  210 01 0 00 057277              MOVN    1,[FMSGCD (SET FNAME file ext <CR> or SET FNAME <CR>)]
035631  260 17 0 00 035623*             GO      .SARG                   ; get argument (extension)
035632  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
035633  254 00 0 00 035636               JRST   DSETF0                  ; no extension given - exit
035634  200 00 0 00 035626*             MOVE    ARGUM                   ; get 2nd argument
035635  202 00 0 00 000000#             MOVEM   UNAME+1                 ; save it
035636  200 00 0 00 064645'     DSETF0: MOVE    TMP1                    ; get first argument
035637  202 00 0 00 000000*             MOVEM   UNAME                   ; save it
035640  263 17 0 00 000000              RTN                             ; return

035641  037 00 0 00 030242      DSETF1: PCRL
035642  037 00 0 00 057300      DSETF2: TMSG    <Microcode file name ....... >
035643  200 00 0 00 035637*             MOVE    UNAME                   ; get name
035644  037 00 0 00 000002              PNTSIX                          ; print it
035645  037 00 0 00 057306              TMSG    <.>
035646  200 00 0 00 000000#             MOVE    UNAME+1                 ; get extension
035647  037 00 0 00 000002              PNTSIX                          ; print it
035650  037 00 0 00 030242              PCRL
035651  263 17 0 00 000000              RTN                             ; return
```

```
                                     ;#***************************************************************
                                     ;* DSETWH - Print all selected options
                                     ;#***************************************************************

035652  260 17 0 00 035603*          DSETWH: GO      LASARG                  ; check for last argument
035653  263 17 0 00 000000                   RTN                             ; error - exit
                                             JRST    [FMSGCD (SET WHAT <CR>) ; ? typed - print msg and
035654  254 00 0 00 057313                            RTN]                   ;   return
035655  037 00 0 00 057315                   TMSGCD  <Options:>
035656  260 17 0 00 035457                   GO      DSETA1                  ; print address
035657  260 17 0 00 035475                   GO      DSETC2                  ; print CSR data
035660  260 17 0 00 035514                   GO      DSETP2                  ; print parity flag
035661  260 17 0 00 035534                   GO      DSETE2                  ; print EBUF flag
035662  260 17 0 00 035555                   GO      DSETL2                  ; print length
035663  260 17 0 00 035610                   GO      DSETD2                  ; print data pattern
035664  260 17 0 00 035642                   GO      DSETF2                  ; print file name selected
035665  263 17 0 00 000000                   RTN                             ; return
```

```
                                          ;#*******************************************************************
                                          ;* DSETFA - Set failing network
                                          ;#*******************************************************************

                                          DSETFA: MOVE     1,[FMSGCD (SET FAULT network # <CR> or SET FAULT <CR>
                                          Where 1-30 = E1-E30
                                                31-53 = M1-M23
                                                54-103 = C1-C30)]
035666  200 01 0 00 057347                        GO       .OARG                     ; get argument (network number)
035667  260 17 0 00 035576*                        RTN                               ; error/altmode/question - exit
035670  263 17 0 00 000000                          JRST   DSETB1                    ; no argument - print it out
035671  254 00 0 00 035700                        MOVE     1,ARGUM                   ; get argument
035672  200 01 0 00 035634*                       CAIL     1,1                       ; in range 1-103 octal?
035673  301 01 0 00 000001                        CAILE    1,103
035674  303 01 0 00 000103                        JRST     [TMSGCD <? Network number out of range 1-103>
                                                            GO      FIOFF
                                                            RTN]
035675  254 00 0 00 057360
035676  202 01 0 00 047032                        MOVEM    1,FLTNUM                  ; save network number
035677  263 17 0 0u 000000                        RTN                                ; return

035700  037 00 0 00 057363                DSETB1: TMSGC    <Failing network ........... >
035701  200 00 0 00 047032                        MOVE     FLTNUM                    ; get network number
035702  037 16 0 00 000003                        PNTOCS                             ; print it
035703  200 01 0 00 047032                        MOVE     1,FLTNUM                  ; get network number
035704  307 01 0 00 000030                        CAIG     1,30                      ; EBUS network?
                                                  PJRST    [TMSG    < - E>           ; yes - print it
                                                            MOVE    1
                                                            PNTOCS
                                                            PCRL
035705  254 00 0 00 057373                                  RTN]
035706  307 01 0 00 000053                        CAIG     1,53                      ; MPROC network?
                                                  PJRST    [TMSG    < - M>           ; yes - print it
                                                            MOVE    1
                                                            SUBI    30
                                                            PNTOCS
                                                            PCRL
                                                            RTN]
035707  254 00 0 00 057401
035710  037 00 0 00 057407                        TMSG     < - C>                    ; print CBUS network
035711  200 00 0 00 000001                        MOVE     1
035712  275 00 0 00 000053                        SUBI     53
035713  037 16 0 00 000003                        PNTOCS
035714  037 00 0 00 030242                        PCRL
035715  263 17 0 00 000000                        RTN                                ; return
```

```
                                        ;#*********************************************************************
                                        ;* DSETPL - List data patterns
                                        ;#*********************************************************************

035716  200 01 0 00 057421      DSETPL: MOVE    1,[FMSGCD <SET PATLIS num1,num2 or SET PATLIS (CR)>]
035717  260 17 0 00 035667*             GO      .OARG                   ; get argument
035720  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
                                          JRST  [MOVE   PAT,DSPAT       ; no argument - just print
                                                 MOVEM  PAT,PAT1#       ;    current selection
                                                 MOVEM  PAT,PAT2#
035721  254 00 0 00 057422                       JRST   DSETPP]
035722  200 00 0 00 035672*             MOVE    ARGUM                   ; get argument
035723  202 00 0 00 064623'             MOVEM   PAT1#                   ; save it

035724  200 01 0 00 057421              MOVE    1,[FMSGCD <SET PATLIS num1,num2 or SET PATLIS (CR)>]
035725  260 17 0 00 035717*             GO      .OARG                   ; get argument
035726  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
                                          JRST  [MOVE   PAT,PAT1        ; no 2nd argument - just print
                                                 MOVEM  PAT,PAT2        ;   one translation
035727  254 00 0 00 057426                       JRST   DSETPP]
035730  200 00 0 00 035722*             MOVE    ARGUM                   ; get argument
035731  202 00 0 00 064624'             MOVEM   PAT2                    ; save it
035732  200 01 0 00 064624'             MOVE    1,PAT2                  ; get argument
035733  274 01 0 00 064623'             SUB     1,PAT1                  ; see if range ok
                                        JUMPL   1,[FMSGCD <? Range nonsensical>
                                                 GO     FIOFF
035734  321 01 0 00 057436                       RTN]

035735  200 14 0 00 064623'     DSETPP: MOVE    PAT,PAT1                ; get pattern
035736  301 14 0 00 000001              CAIL    PAT,1                   ; in range?
035737  303 14 0 00 000126              CAILE   PAT,126
                                        JRST    [FMSGCD <? Out of range 1-126 octal>
                                                 GO     FIOFF
035740  254 00 0 00 057245                       RTN]
035741  037 00 0 00 030242              PCRL
035742  256 00 0 14 035612*             XCT     PATPNT(PAT)             ; print it
035743  350 00 0 00 000014              AOS     PAT                     ; increment pattern number
035744  037 07 0 00 000003              TTALTM                          ; altmode typed?
035745  334 00 0 00 000000              SKIPA                           ; no - continue
035746  254 00 0 00 035751              JRST    DSETPX                  ; yes - return
035747  317 14 0 00 064624'             CAMG    PAT,PAT2                ; done yet?
035750  254 00 0 00 035736              JRST    DSETPP+1                ; no - loop till done
035751  037 00 0 00 030243      DSETPX: PCRL2                           ; final CRLF
035752  263 17 0 00 000000              RTN                             ; exit
```

```
                                        ;#*********************************************************************
                                        ;* DSETHE - Set help message
                                        ;#*********************************************************************

035753  200 01 0 00 057445              DSETHE: MOVE    1,[FMSGCD (SET HELP <CR>)]
035754  260 17 0 00 034366*                     GO      CHKARG                  ; check for argument
035755  263 17 0 00 000000                      RTN                             ; error - exit
035756  037 01 0 00 000026                      PFORCE                          ; handle Control-O
035757  037 00 1 00 035761                      PNTMSG  @DSETHL                 ; print text
035760  263 17 0 00 000000                      RTN                             ; exit

035761                                  DSETHL: [ASCIZ  ^
                                        SET Command:

                                        SET ADDR adr    - set start address
                                        SET CSR data    - set start CSR data
                                        SET PARITY arg* - set/clr auto par generation
                                        SET EBUF arg*   - set/clr preserve EBUF flag
                                        SET LENGTH len  - set buffer length
                                        SET PAT n       - set data pattern
                                        SET FNAME name ext - set default ucode file name
                                        SET PATLIS arg  - list data patterns
                                        SET WHAT        - print options
                                        SET HELP        - print this

                                        * Arg - YES or NO
035761  000000  057446                  ^]


                                        ;#*********************************************************************
                                        ;* Program Flags
                                        ;#*********************************************************************

035762  000000  000000                  DSADDR: 0                               ; start address initially zero
035763  000000  000000                  DSDATA: 0                               ; start data initially zero

035764  777777  777777                  PARFLG: -1                              ; 'automatic parity generation' flag
035765  777777  777777                  PRSFLG: -1                              ; 'preserve EBUF' flag

035766  000000  000200                  DSWORD: ^D128                           ; buffer length

035767  000000  000001                  DSPAT:  1                               ; data pattern
035770  000000  000000                  DSTPAT: 0                               ; operator select pattern
```

```

                                                ;#*********************************************************************
                                                ;* DECSR - Examine CSR register
                                                ;#*********************************************************************

035771  200 01 0 00 057600              DECSR:  MOVE    1,[FMSGCD (ECSR <CR>)]
035772  260 17 0 00 035754*                     GO      CHKARG                          ; check for argument
035773  263 17 0 00 000000                      RTN                                     ; error - exit
035774  260 17 0 00 032341*                     GO      RDCSR                           ; read the CSR register
035775  037 00 0 00 057601                        TMSGC <[Rdcsr failed]>                ; error return
035776  260 17 0 00 043715                      GO      CSRPNT                          ; print CSR data
035777  037 00 0 00 030242                      PCRL                                    ; end of line
036000  263 17 0 00 000000                      RTN                                     ; return


                                                ;#*********************************************************************
                                                ;* DDCSR - Deposit CSR register
                                                ;#*********************************************************************

036001  210 01 0 00 057612              DDCSR:  MOVN    1,[FMSGCD (DCSR xxxxxx <CR>)]
036002  260 17 0 00 035725*                     GO      .OARG                           ; get argument
036003  263 17 0 00 000000                        RTN                                   ; error/altmode/question - exit
                                                  JRST  [FMSGCD <? Missing argument>
                                                          GO      FIOFF
036004  254 00 0 00 055164                                RTN]
036005  200 01 0 00 035730*                     MOVE    1,ARGUM                         ; get argument
036006  260 17 0 00 000000*                     GO      LDCSR                           ; write it
036007  263 17 0 00 000000                      RTN                                     ; return


                                                ;#*********************************************************************
                                                ;* DZCSR - Zero CSR register
                                                ;#*********************************************************************

036010  200 01 0 00 057616              DZCSR:  MOVE    1,[FMSGCD (ZCSR <CR>)]
036011  260 17 0 00 035772*                     GO      CHKARG                          ; check for argument
036012  263 17 0 00 000000                      RTN                                     ; error - exit
036013  400 01 0 00 000000                      SETZ    1,                              ; clear data
036014  260 17 0 00 036006*                     GO      LDCSR                           ; write it
036015  263 17 0 00 000000                      RTN                                     ; return


                                                ;#*********************************************************************
                                                ;* DCONI - Loop on CONI's
                                                ;#*********************************************************************

036016  200 01 0 00 057622              DCONI:  MOVE    1,[FMSGCD (LCONI <CR>)]
036017  260 17 0 00 036011*                     GO      CHKARG                          ; check for argument
036020  263 17 0 00 000000                      RTN                                     ; error - exit
036021  260 17 0 00 035774*             DCONI1: GO      RDCSR                           ; read the CSR register
036022  255 00 0 00 000000                        JFCL                                  ; error return
036023  037 07 0 00 000003                      TTALTM                                  ; altmode typed?
036024  254 00 0 00 036021                      JRST    DCONI1                          ; no - keep looping
036025  263 17 0 00 000000                      RTN                                     ; return
```

```
                                                ;#*********************************************************************
                                                ;* DCONO - Loop on CONO's
                                                ;#*********************************************************************

036026  210 01 0 00 057630      DCONO:  MOVN    1,[FMSGCD (LCONO xxxxxx <CR>)]
036027  260 17 0 00 036002*             GO      .OARG                   ; get argument
036030  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
                                         JRST   [FMSGCD <? Missing argument>
                                                 GO     FIOFF
036031  254 00 0 00 055164                       RTN]
036032  200 01 0 00 036005*     DCONO1: MOVE    1,ARGUM                 ; get argument
036033  260 17 0 00 036014*             GO      LDCSR                   ; write it
036034  037 07 0 00 000003              TTALTM                          ; altmode typed?
036035  254 00 0 00 036032              JRST    DCONO1                  ; no - keep looping
036036  263 17 0 00 000000              RTN                             ; return


                                                ;#*********************************************************************
                                                ;* DDATI - Loop on DATAI's
                                                ;#*********************************************************************

036037  200 01 0 00 057635      DDATI:  MOVE    1,[FMSGCD (LDATAI <CR>)]
036040  260 17 0 00 036017*             GO      CHKARG                  ; check for argument
036041  263 17 0 00 000000              RTN                             ; error - exit
036042  260 17 0 00 000000*     DDATI1: GO      RDEBUF                  ; do a DATAI
036043  037 07 0 00 000003              TTALTM                          ; altmode typed?
036044  254 00 0 00 036042              JRST    DDATI1                  ; no - keep looping
036045  263 17 0 00 000000              RTN                             ; return


                                                ;#*********************************************************************
                                                ;* DDATO - Loop on DATAO's
                                                ;#*********************************************************************

036046  210 01 0 00 057643      DDATO:  MOVN    1,[FMSGCD (LDATAO xxxxxx <CR>)]
036047  260 17 0 00 036027*             GO      .OARG                   ; get argument
036050  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
                                         JRST   [FMSGCD <? Missing argument>
                                                 GO     FIOFF
036051  254 00 0 00 055164                       RTN]
036052  200 01 0 00 036032*     DDATO1: MOVE    1,ARGUM                 ; get argument
036053  260 17 0 00 000000*             GO      LDEBUF                  ; do a DATAO
036054  037 07 0 00 000003              TTALTM                          ; altmode typed?
036055  254 00 0 00 036052              JRST    DDATO1                  ; no - keep looping
036056  263 17 0 00 000000              RTN                             ; return
```

```



                                        ;#*********************************************************************
                                        ;* DROUTN - Routine to do special diagnostic loop
                                        ;#*********************************************************************

036057  210 01 0 00 057652              DROUTN: MOVN    1,[FMSGCD (LROUTN xxxxxxxxxxxx <CR>)]
036060  260 17 0 00 036047*                     GO      .OARG                   ; get argument
036061  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
036062  255 00 0 00 000000                       JFCL
036063  260 17 0 00 000000*             DROUT1: GO      IPACLR                  ; do a 'port clear'
036064  255 00 0 00 000000                      JFCL
036065  255 00 0 00 000000                      JFCL
036066  255 00 0 00 000000                      JFCL
036067  255 00 0 00 000000                      JFCL
036070  255 00 0 00 000000                      JFCL
036071  255 00 0 00 000000                      JFCL
036072  255 00 0 00 000000                      JFCL
036073  255 00 0 00 000000                      JFCL
036074  255 00 0 00 000000                      JFCL
036075  255 00 0 00 000000                      JFCL
036076  255 00 0 00 000000                      JFCL
036077  255 00 0 00 000000                      JFCL
036100  255 00 0 00 000000                      JFCL
036101  255 00 0 00 000000                      JFCL
036102  255 00 0 00 000000                      JFCL
036103  037 07 0 00 000003                      TTALTM                          ; check for altmode typed
036104  254 00 0 00 036063                      JRST    DROUT1                  ; no - keep looping
036105  263 17 0 00 000000                      RTN                             ; yes - return



                                        ;#*********************************************************************
                                        ;* DRESET - Issue an EBUS Reset
                                        ;#*********************************************************************

036106  200 01 0 00 057656              DRESET: MOVE    1,[FMSGCD (RESET <CR>)]
036107  260 17 0 00 036040*                     GO      CHKARG                  ; check for argument
036110  263 17 0 00 000000                      RTN                             ; error - exit
036111  260 17 0 00 000000*                     GO      ERESET
036112  263 17 0 00 000000                      RTN                             ; return



                                        ;#*********************************************************************
                                        ;* DCLEAR - Issue a 'Port Clear'
                                        ;#*********************************************************************

036113  200 01 0 00 057662              DCLEAR: MOVE    1,[FMSGCD (CLEAR <CR>)]
036114  260 17 0 00 036107*                     GO      CHKARG                  ; check for argument
036115  263 17 0 00 000000                      RTN                             ; error - exit
036116  260 17 0 00 036063*                     GO      IPACLR                  ; do a 'port clear'
036117  263 17 0 00 000000                      RTN                             ; return
```

```
                                      ;#*******************************************************************
                                      ;* DSINIT - Set up start addr/data/history prior to single step command
                                      ;#*******************************************************************

036120  200 01 0 00 057667            DSINIT: MOVE    1,[FMSGCD (SSINIT <CR>)]
036121  260 17 0 00 036114*                   GO      CHKARG                  ; check for argument
036122  263 17 0 00 000000                    RTN                             ; error - exit
036123  200 00 0 00 035762                    MOVE    DSADDR                  ; get start address
036124  202 00 0 00 000000*                   MOVEM   SNEXT                   ; save it for IPASST
036125  200 00 0 00 035763                    MOVE    DSDATA                  ; get start CSR data
036126  202 00 0 00 000000*                   MOVEM   SDATA                   ; save it for IPASST
036127  260 17 0 00 035457                    GO      DSETA1                  ; print start address
036130  260 17 0 00 035475                    GO      DSETC2                  ; print start data
036131  260 17 0 00 036274                    GO      DSSCLR                  ; clear sstep history
036132  263 17 0 00 000000                    RTN
```

```
                                                 ;#***************************************************************************
                                                 ;* DSTART - Start the port
                                                 ;
                                                 ; Arguments set up:
                                                 ;
                                                 ; DSADDR - Current start address which has been specified by either a
                                                 ;          'SETADD' command, or as an argument to the 'START' command.
                                                 ; DSDATA - Current start CSR data specified by a 'SETCSR' command.
                                                 ;
                                                 ; Arguments set up for IPASST:
                                                 ;
                                                 ; SNEXT - Contents of DSADDR is used.
                                                 ; SLAST - Contents of DSADDR is used.
                                                 ; SDATA - Contents of DSDATA is used.
                                                 ;#***************************************************************************

036133  210 01 0 00 057677      DSTART: MOVN    1,[FMSGCD (START addr <CR> or START <CR>)]
036134  260 17 0 00 036060*             GO      .OARG                   ; get start address
036135  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
036136  334 01 0 00 035762               SKIPA  1,DSADDR                ; no argument given

                                ; Start it up

036137  260 17 0 00 000000*     DSTAR0: GO      ISTOP                   ; stop the port
036140  200 01 0 00 036052*             MOVE    1,ARGUM                 ; get argument
036141  202 01 0 00 036124*             MOVEM   1,SNEXT                 ; set it up
036142  202 00 0 00 000000*             MOVEM   SLAST                   ; set it up
036143  200 00 0 00 035763              MOVE    DSDATA                  ; get start data
036144  202 00 0 00 036126*             MOVEM   SDATA                   ; set it up
036145  260 17 0 00 000000*             GO      IPASRT                  ; start it up
                                         JRST   [FMSGCD <? Error accessing CSR register>
                                                 GO     FIOFF
036146  254 00 0 00 057707                       RTN]
                                         JRST   [FMSGCD <? Port not stopped>
                                                 GO     FIOFF
036147  254 00 0 00 057717                       RTN]
036150  334 00 0 00 000000               SKIPA                          ; error - Error bits set in CSR
036151  263 17 0 00 000000              RTN                             ; exit
036152  037 01 0 00 057722              FMSGCD  <? START failed - Error bits set in CSR>
036153  260 17 0 00 036021*             GO      RDCSR                   ; read CSR register
036154  255 00 0 00 000000              JFCL                            ; ignore error
036155  260 17 0 00 043715              GO      CSRPNT                  ; print CSR register
036156  037 00 0 00 030242              PCRL                            ; end of line
036157  263 17 0 00 000000              RTN                             ; continue
```

```

                                                ;#*********************************************************************
                                                ;* DSTOP - Stop the port
                                                ;
                                                ; Arguments set by IPASTP:
                                                ;
                                                ; SNEXT - Set to stopping address.
                                                ; SDATA - Set to contents of CSR register.
                                                ;#*********************************************************************

036160  200 01 0 00 057736                      DSTOP:  MOVE    1,[FMSGCD (STOP <CR>)]
036161  260 17 0 00 036121*                             GO      CHKARG                          ; check for argument
036162  263 17 0 00 000000                              RTN                                     ; error - exit
036163  260 17 0 00 000000*                             GO      IPASTP                          ; stop it and get final address
                                                         JRST   [FMSGCD <? Error accessing CSR register>
                                                                 GO     FIOFF
036164  254 00 0 00 057707                                       RTN]
                                                         JRST   [FMSGCD <? Port not running>
                                                                 GO     FIOFF
036165  254 00 0 00 057744                                       RTN]
036166  254 00 0 00 036173                               JRST   DSTOP1                          ; error - Error bits set in CSR
036167  037 00 0 00 057747                              TMSGC   <Port stopped.  Last Adr = >
036170  200 00 0 00 036141*                             MOVE    SNEXT                           ; print it out
036171  037 04 0 00 000000                              PNT4
036172  334 00 0 00 000000                              SKIPA
036173  037 01 0 00 057755                      DSTOP1: FMSGCD  <? Stop failed - Error bits set in CSR>
036174  200 01 0 00 036144*                             MOVE    1,SDATA                         ; get CSR data
036175  260 17 0 00 043715                              GO      CSRPNT                          ; print CSR register
036176  037 00 0 00 030242                              PCRL                                    ; end of line
036177  263 17 0 00 000000                              RTN                                     ; continue
```

```
                                        ;#*********************************************************************
                                        ;* DSSTEP - Single step the port
                                        ;* DTRACE - Single step the port and trace progress
                                        ;
                                        ; Arguments set up for IPASST:
                                        ;
                                        ; SNEXT - This is the address last set up by another single step
                                        ;         command or by a SETSTR (set up start data) command.
                                        ;         After each single step the LAR is read.  The contents
                                        ;         of the LAR are put in SNEXT.
                                        ;
                                        ; SLAST - This is the address last executed.  SNEXT is put in this
                                        ;         location before each single step
                                        ;
                                        ; SDATA - This is the data last set up by another single step
                                        ;         command or by a SETSTR (set up start data) command.
                                        ;
                                        ; Both of these routines store single step data in a ring buffer which
                                        ; can be examined with a SPRINT command.  This is useful in debugging
                                        ; a microcode routine and instead of single stepping and tracing, you
                                        ; single step 500 times or until an error occurs, then examine last x
                                        ; locations to see what happened.
                                        ;#*********************************************************************

036200  210 01 0 00 057777              DTRACE: MOVN    1,[FMSGCD (STRACE RepeatCount <CR> or STRACE <CR>)]
036201  476 00 0 00 064577'                     SETOM   DTRFLG                  ; set trace flag
036202  254 00 0 00 036205                      JRST    DTRACO                  ; continue

036203  402 00 0 00 064577'             DSSTEP: SETZM   DTRFLG#                 ; clear trace flag
036204  210 01 0 00 060011                      MOVN    1,[FMSGCD (SSTEP RepeatCount <CR> or SSTEP <CR>)]

036205  402 00 0 00 036140*             DTRACO: SETZM   ARGUM                   ; set default argument
036206  350 00 0 00 036205*                     AOS     ARGUM                   ;   to 1 single step
036207  260 17 0 00 035543*                     GO      .DARG                   ; get argument
036210  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
036211  255 00 0 00 000000                       JFCL                           ; no argument given
036212  260 17 0 00 036137*                     GO      ISTOP                   ; stop the port
036213  200 02 0 00 036206*                     MOVE    2,ARGUM                 ; get argument
036214  400 04 0 00 000000                      SETZ    4,                      ; clear count of single steps
036215  332 00 0 00 064577'                     SKIPE   DTRFLG                  ; trace flag set?
036216  037 00 0 00 060012                      TMSGCD  < RAR   LAR   CSR (final)  SS#>

                                        ; Single step

036217  350 00 0 00 000004              DSSTO:  AOS     4                       ; increment count
036220  200 01 0 00 035765                      MOVE    1,PRSFLG                ; get preserve EBUF flag
036221  200 00 0 00 036174*                     MOVE    SDATA                   ; get initial CSR data
036222  202 00 0 00 064642'                     MOVEM   SLDATA#                 ; save initial CSR data
```

```
036223  260 17 0 00 000000*            GO      IPASST                   ; single step once
                                        JRST   [FMSGCD <? Error accessing CSR register>
                                                GO     FIOFF
036224  254 00 0 00 057707                      RTN]
                                        JRST   [FMSGCD <? Port not stopped>
                                                GO     FIOFF
036225  254 00 0 00 057717                      RTN]
                                        JRST   [GO     SAVHST
                                                TMSGC  <[Error bits set in CSR]>
                                                SKIPE  DTRFLG
                                                JRST   DSSTX
                                                TMSGC  <[>
                                                MOVE   4
                                                PNTOCS
                                                TMSGD  < ssteps done]>
036226  254 00 0 00 060034                      JRST   DSSTX]
036227  260 17 0 00 037523             GO      SAVHST                   ; save data for sstep history

                               ; If tracing is being done

036230  200 01 0 00 036303             MOVE    1,SSLOC                  ; get history location
036231  332 00 0 00 064577'            SKIPE   DTRFLG                   ; tracing being done?
036232  260 17 0 00 037504             GO      PNTHST                   ; yes - print it

                               ; Altmode and exit checking

036233  037 07 0 00 000003     DSST1:  TTALTM                           ; altmode key struck?
036234  334 00 0 00 000000             SKIPA                            ; no - proceed
                                       JRST     [PCRL                   ; yes - print some final
                                                 SKIPE  DTRFLG          ;    data and exit
                                                 RTN
                                                 TMSG   <[>
                                                 MOVE   4
                                                 PNTOCS
                                                 TMSGD  < ssteps done]>
036235  254 00 0 00 060046                       JRST   DSSTX]
036236  367 02 0 00 036217             SOJG     2,DSSTO                 ; loop till done

                               ; Exit

036237  037 00 0 00 060056     DSSTX:  TMSGC    <LAR Last/>             ; print LAR data
036240  200 00 0 00 036142*            MOVE     SLAST                   ; get last address
036241  037 04 0 00 000000             PNT4                             ; print it
036242  037 00 0 00 060061             TMSG     <  Next/>
036243  200 00 0 00 036170*            MOVE     SNEXT                   ; get next address
036244  037 04 0 00 000000             PNT4                             ; print it
036245  037 00 0 00 060063             TMSG     <  CSR/>                ; print CSR data
036246  200 00 0 00 036221*            MOVE     SDATA                   ; get it
036247  037 13 0 00 000000             PNTHW                            ; print it
036250  037 00 0 00 030242             PCRL                             ; end of line
036251  263 17 0 00 000000             RTN
```

```
                                        ;#*********************************************************************
                                        ;* DSSPNT - Print single step history
                                        ;#*********************************************************************

036252  201 00 0 00 000024              DSSPNT: MOVEI   ^D20                            ; set default number of ssteps
036253  202 00 0 00 036213*                     MOVEM   ARGUM                           ;   to print to 20
036254  210 01 0 00 060075                      MOVN    1,[FMSGCD (SPRINT EntryCount or SPRINT <CR>)]
036255  260 17 0 00 036207*                     GO      .DARG                           ; get argument
036256  263 17 0 00 000000                       RTN                                    ; error/altmode/question - exit
036257  255 00 0 00 000000                       JFCL                                   ; no argument given
036260  200 03 0 00 036253*                     MOVE    3,ARGUM                         ; get argument
036261  200 01 0 00 036303                      MOVE    1,SSLOC                         ; get current location
036262  037 00 0 00 060012                      TMSGCD  < RAR   LAR   CSR (final)  SS#>
036263  260 17 0 00 037504              DSSPNO: GO      PNTHST                          ; print an item
036264  037 07 0 00 000003                      TTALTM                                  ; altmode typed?
036265  334 00 0 00 000000                      SKIPA                                   ; no - continue
036266  263 17 0 00 000000                      RTN                                     ; yes - exit
036267  375 00 0 00 000001                      SOSGE   1                               ; decrement pointer - at start of list?
036270  201 01 0 00 000177                      MOVEI   1,177                           ; yes - point to end now
036271  333 00 0 01 036304                      SKIPLE  SSNUM(1)                        ; valid entry?
036272  367 03 0 00 036263                      SOJG    3,DSSPNO                        ; yes - loop till done
036273  263 17 0 00 000000                      RTN                                     ; no - return


                                        ;#*********************************************************************
                                        ;* DSSCLR - Clear single step history data
                                        ;#*********************************************************************

036274  200 01 0 00 060102              DSSCLR: MOVE    1,[FMSGCD (SCLEAR <CR>)]
036275  260 17 0 00 036161*                     GO      CHKARG                          ; check for argument
036276  263 17 0 00 000000                      RTN                                     ; error - exit
036277  402 00 0 00 036303                      SETZM   SSLOC                           ; clear first word
036300  200 00 0 00 060103                      MOVE    [SSLOC,,SSLOC+1]                ; build BLT pointer
036301  251 00 0 00 037503                      BLT     SSCSRN+177                      ; clear it
036302  263 17 0 00 000000                      RTN                                     ; return

                                        ; Ring buffer and miscellaneous data

036303  000000  000000                  SSLOC:  0                                       ; pointer to single step entry
036304                                  SSNUM:  BLOCK   200                             ; latest single step number
036504                                  SSADRL: BLOCK   200                             ; initial address
036704                                  SSADRN: BLOCK   200                             ; final address
037104                                  SSCSRL: BLOCK   200                             ; initial CSR
037304                                  SSCSRN: BLOCK   200                             ; final CSR
```

```
                                              ;#*********************************************************************
                                              ;* PNTHST - Print a single step history entry
                                              ;
                                              ; Argument:     AC1 - location to print (offset into tables)
                                              ;#*********************************************************************

037504  261 17 0 00 000000                    PNTHST: PUT     0                       ; save AC0
037505  200 00 0 01 036504                            MOVE    SSADRL(1)               ; print LAR (last) contents
037506  037 04 0 00 000000                            PNT4
037507  037 00 0 00 053077                            TMSG    < >
037510  200 00 0 01 036704                            MOVE    SSADRN(1)               ; print LAR (next) contents
037511  037 04 0 00 000000                            PNT4
037512  037 00 0 00 053077                            TMSG    < >
037513  200 00 0 01 037304                            MOVE    SSCSRN(1)               ; get CSR (final) data
037514  037 13 0 00 000000                            PNTHW                           ; print it
037515  037 00 0 00 053077                            TMSG    < >
037516  200 00 0 01 036304                            MOVE    SSNUM(1)                ; get sstep number
037517  037 16 0 00 000003                            PNTOCS                          ; print it
037520  037 00 0 00 030242                            PCRL
037521  262 17 0 00 000000                            GET     0                       ; restore AC0
037522  263 17 0 00 000000                            RTN                             ; return



                                              ;#*********************************************************************
                                              ;* SAVHST - Save single step history data
                                              ;#*********************************************************************

037523  261 17 0 00 000000                    SAVHST: RPUT    (0,1,2)                 ; save AC's
037526  200 02 0 00 036303                            MOVE    2,SSLOC                 ; get last offset
037527  200 02 0 02 036304                            MOVE    2,SSNUM(2)              ; get last number
037530  350 01 0 00 036303                            AOS     1,SSLOC                 ; get/increment offset
037531  301 01 0 00 000200                            CAIL    1,200                   ; overflow ring buffer yet?
037532  403 01 0 00 036303                            SETZB   1,SSLOC                 ; yes - reinitialize pointer
037533  350 00 0 00 000002                            AOS     2                       ; increment ss number
037534  202 02 0 01 036304                            MOVEM   2,SSNUM(1)              ; save it
037535  200 00 0 00 036240*                           MOVE    SLAST                   ; get initial address
037536  202 00 0 01 036504                            MOVEM   SSADRL(1)               ; save it
037537  200 00 0 00 036243*                           MOVE    SNEXT                   ; get final address
037540  202 00 0 01 036704                            MOVEM   SSADRN(1)               ; save it
037541  200 00 0 00 064642'                           MOVE    SLDATA                  ; get initial CSR data
037542  202 00 0 01 037104                            MOVEM   SSCSRL(1)               ; save it
037543  200 00 0 00 036246*                           MOVE    SDATA                   ; get final CSR data
037544  202 00 0 01 037304                            MOVEM   SSCSRN(1)               ; save it
037545  262 17 0 00 000002                            RGET    (2,1,0)                 ; restore AC's
037550  263 17 0 00 000000                            RTN                             ; return
```

```
                                             ;#*********************************************************************
                                             ;* DCONT - Continue the port
                                             ;
                                             ; Arguments set up for IPASRT:
                                             ;
                                             ; SNEXT - Points to next address, from IPASTP or IPASST
                                             ; SDATA - Defaults to current contents of SDATA last set up by IPASTP.
                                             ;#*********************************************************************

037551  200 01 0 00 060107                   DCONT:  MOVE    1,[FMSGCD (CONT <CR>)]
037552  260 17 0 00 036275*                          GO      CHKARG                  ; check for argument
037553  263 17 0 00 000000                           RTN                             ; error - exit
037554  260 17 0 00 036145*                          GO      IPASRT                  ; start it up
                                                      JRST   [FMSGCD <? Error accessing CSR register>
                                                              GO      FIOFF
037555  254 00 0 00 057707                                    RTN]
                                                      JRST   [FMSGCD <? Port not stopped>
                                                              GO      FIOFF
037556  254 00 0 00 057717                                    RTN]
037557  334 00 0 00 000000                            SKIPA                          ; error - Error bits set in CSR
037560  263 17 0 00 000000                           RTN                             ; exit
037561  037 01 0 00 060110                           FMSGCD  <? CONT failed - Error bits set in CSR>
037562  200 01 0 00 037543*                          MOVE    1,SDATA                 ; get CSR data
037563  260 17 0 00 043715                           GO      CSRPNT                  ; print CSR register
037564  037 01 0 00 030242                           PCRLF                           ; end of line
037565  263 17 0 00 000000                           RTN                             ; continue


                                             ;#*********************************************************************
                                             ;* DEEBUF - Examine EBUF register
                                             ;#*********************************************************************

037566  200 01 0 00 060124                   DEEBUF: MOVE    1,[FMSGCD (EEBUF <CR>)]
037567  260 17 0 00 037552*                          GO      CHKARG                  ; check for argument
037570  263 17 0 00 000000                           RTN                             ; error - exit
037571  260 17 0 00 036212*                          GO      ISTOP                   ; stop the port
037572  260 17 0 00 000000*                          GO      SETEBU                  ; ensure 'Test EBUF' bit is set
                                                      JRST   [FMSGCD <? Error accessing CSR register>
                                                              GO      FIOFF
037573  254 00 0 00 057707                                    RTN]

                                             ; Now read the data

037574  260 17 0 00 036042*                          GO      RDEBUF                  ; read EBUF
037575  037 00 0 00 060125                           TMSGC   <EBUF/ >                ; print out the data
037576  200 00 0 00 000001                           MOVE    1
037577  037 13 0 00 000000                           PNTHW
037600  037 00 0 00 030242                           PCRL
037601  263 17 0 00 000000                           RTN                             ; continue
```

```
                                         ;#***************************************************************
                                         ;* DDEBUF - Deposit EBUF register
                                         ;* DZEBUF - Zero EBUF register
                                         ;#***************************************************************

037602  200 01 0 00 060132               DZEBUF: MOVE    1,[FMSGCD (ZEBUF <CR>)]
037603  260 17 0 00 037567*                      GO      CHKARG                  ; check for argument
037604  263 17 0 00 000000                       RTN                             ; error - exit
037605  402 00 0 00 036260*                      SETZM   ARGUM                   ; clear data
037606  254 00 0 00 037613                       JRST    DDEBU1                  ; continue

037607  210 01 0 00 060141               DDEBUF: MOVN    1,[FMSGCD (DEBUF xxxxxxxxxxxx <CR>)]
037610  260 17 0 00 036134*                      GO      .OARG                   ; get argument
037611  263 17 0 00 000000                        RTN                            ; error/altmode/question - exit
                                                  JRST   [FMSGCD <? Missing argument>
                                                          GO      FIOFF
037612  254 00 0 00 055164                                RTN]
037613  260 17 0 00 037571*              DDEBU1: GO      ISTOP                   ; stop the port
037614  260 17 0 00 037572*                      GO      SETEBU                  ; ensure 'Test EBUF' bit is set
                                                  JRST   [FMSGCD <? Error accessing CSR register>
                                                          GO      FIOFF
037615  254 00 0 00 057707                                RTN]
037616  200 01 0 00 037605*                      MOVE    1,ARGUM                 ; get argument
037617  260 17 0 00 036053*                      GO      LDEBUF                  ; write it
037620  263 17 0 00 000000                       RTN                             ; continue


                                         ;#***************************************************************
                                         ;* DELAR - Examine LAR register
                                         ;#***************************************************************

037621  200 01 0 00 060145               DELAR:  MOVE    1,[FMSGCD (ELAR <CR>)]
037622  260 17 0 00 037603*                      GO      CHKARG                  ; check for argument
037623  263 17 0 00 000000                       RTN                             ; error - exit
037624  260 17 0 00 037613*                      GO      ISTOP                   ; stop the port
037625  201 01 0 00 040000                       MOVEI   1,SELLAR                ; get 'DIAG Sel LAR' bit
037626  260 17 0 00 036033*                      GO      LDCSR                   ; write to CSR register
037627  260 17 0 00 000000*                      GO      RDLAR                   ; read LAR
037630  037 00 0 00 060146                       TMSGC   <LAR: >                 ; print out the data
037631  200 00 0 00 000001                       MOVE    1
037632  037 16 0 00 000003                       PNTOCS
037633  037 00 0 00 060150                       TMSG    < (Adr: >
037634  242 01 0 00 777777                       LSH     1,-1                    ; don't print out RAR bit
037635  200 00 0 00 000001                       MOVE    1
037636  037 16 0 00 000003                       PNTOCS
037637  037 00 0 00 054233                       TMSGD   <)>
037640  263 17 0 00 000000                       RTN                             ; continue
```

```

                                 ;#*********************************************************************
                                 ;* DDRAR - Deposit RAR register
                                 ;#*********************************************************************

037641  210 01 0 00 060156       DDRAR:  MOVN    1,[FMSGCD (DRAR addr <CR>)]
037642  260 17 0 00 037610*              GO      .OARG                           ; get test number
037643  263 17 0 00 000000                RTN                                    ; error/altmode/question - exit
                                          JRST   [FMSGCD <? Missing argument>
                                                  GO      FIOFF
                                                  RTN]
037644  254 00 0 00 075164
037645  260 17 0 00 037624*              GO      ISTOP                           ; stop the port
037646  400 01 0 00 000000               SETZ    1,                              ; ensure we can write the RAR
037647  260 17 0 00 037626*              GO      LDCSR                           ;   by clearing CSR register
037650  200 01 0 00 037616*              MOVE    1,ARGUM                         ; get argument
037651  260 17 0 00 000000*              GO      LDRAR                           ; write it
037652  263 17 0 00 000000               RTN                                     ; continue


                                 ;#*********************************************************************
                                 ;* DZRAR - Zero RAR register
                                 ;#*********************************************************************

037653  200 01 0 00 060162       DZRAR:  MOVE    1,[FMSGCD (ZRAR <CR>)]
037654  260 17 0 00 037622*              GO      CHKARG                          ; check for argument
037655  263 17 0 00 000000               RTN                                     ; error - exit
037656  260 17 0 00 037645*              GO      ISTOP                           ; stop the port
037657  400 01 0 00 000000               SETZ    1,                              ; clear data
037660  260 17 0 00 037647*              GO      LDCSR                           ; clear CSR first
037661  260 17 0 00 037651*              GO      LDRAR                           ; write it
037662  263 17 0 00 000000               RTN                                     ; continue
```

```
                                  ;#********************************************************************
                                  ;* DECRAM - Examine CRAM locations specified by 'PARG1'' and 'PARG2''.
                                  ;#********************************************************************

037663  201 00 0 00 000000        DECRAM: MOVEI   0                       ; set the 'EDZL' flag
037664  202 00 0 00 064576'               MOVEM   DFLAG#                  ;   to 0 (Examine)
037665  254 00 0 00 037705                JRST    DCOMGO                  ; process arguments

                                  ;#********************************************************************
                                  ;* DDCRAM - Change CRAM locations.
                                  ;#********************************************************************

037666  201 00 0 00 000001        DDCRAM: MOVEI   1                       ; set the 'EDZL' flag
037667  202 00 0 00 064576'               MOVEM   DFLAG                   ;   to 1 (Deposit)
037670  254 00 0 00 037705                JRST    DCOMGO                  ; process arguments

                                  ;#********************************************************************
                                  ;* DBCRAM - Change CRAM locations by field
                                  ;#********************************************************************

037671  201 00 0 00 000002        DBCRAM: MOVEI   2                       ; set the 'EDZL' flag
037672  202 00 0 00 064576'               MOVEM   DFLAG                   ;   to 1 (Deposit)
037673  254 00 0 00 037705                JRST    DCOMGO                  ; process arguments

                                  ;#********************************************************************
                                  ;* DACRAM - Alter CRAM locations.
                                  ;#********************************************************************

037674  201 00 0 00 000003        DACRAM: MOVEI   3                       ; set the 'EDZL' flag
037675  202 00 0 00 064576'               MOVEM   DFLAG                   ;   to 1 (Deposit)
037676  254 00 0 00 037705                JRST    DCOMGO                  ; process arguments

                                  ;#********************************************************************
                                  ;* DZCRAM -- Zero CRAM addresses specified.
                                  ;#********************************************************************

037677  201 00 0 00 000004        DZCRAM: MOVEI   4                       ; set the 'EDZL' flag
037700  202 00 0 00 064576'               MOVEM   DFLAG                   ;   to 2 (Zero)
037701  254 00 0 00 037705                JRST    DCOMGO                  ; process arguments

                                  ;#********************************************************************
                                  ;* DLCRAM -- List CRAM addresses specified.
                                  ;#********************************************************************

037702  201 00 0 00 000005        DLCRAM: MOVEI   5                       ; set the 'EDZL' flag
037703  202 00 0 00 064576'               MOVEM   DFLAG                   ;   to 3 (List)
037704  254 00 0 00 037705                JRST    DCOMGO                  ; process arguments
```

```

                                        ; Start to process the arguments typed - both should be CRAM
                                        ; addresses in octal.

037705  200 00 0 00 037753              DCOMGO: MOVE    RTO                     ; set up default range of CRAM locs
037706  202 00 0 00 037747                      MOVEM   CRAMTO                  ;   to be whatever typed before or
037707  200 00 0 00 037754                      MOVE    RFROM                   ;   if nothing typed, then 0-0
037710  202 00 0 00 037746                      MOVEM   CRAMFR

                                        ; Get first address

037711  200 01 0 00 060170                      MOVE    1,[FMSGCD <xCRAM addr,addr (CR)>]
037712  260 17 0 00 037642*                     GO      .OARG                   ; get first address
037713  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
037714  254 00 0 00 037726                       JRST   DCOMO                   ; no arguments given - continue
037715  200 00 0 00 037650*                     MOVE    ARGUM                   ; get argument
037716  202 00 0 00 037746                      MOVEM   CRAMFR                  ; save in 'from' location
037717  202 00 0 00 037747                      MOVEM   CRAMTO                  ; save in 'to' location

                                        ; Get 2nd address

037720  210 01 0 00 060170                      MOVN    1,[FMSGCD <xCRAM addr,addr (CR)>]
037721  260 17 0 00 037712*                     GO      .OARG                   ; get first address
037722  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
037723  254 00 0 00 037726                       JRST   DCOMO                   ; no arguments given - continue
037724  200 00 0 00 037715*                     MOVE    ARGUM                   ; get argument
037725  202 00 0 00 037747                      MOVEM   CRAMTO                  ; save in 'to' location

                                        ; Now range check the arguments

037726  331 01 0 00 037746              DCOMO:  SKIPL   1,CRAMFR                ; 1st argument in range?
037727  303 01 0 00 007777                      CAILE   1,7777
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         SETOM  ALTF
                                                         GO     FIOFF
037730  254 00 0 00 060171                               RTN]
037731  331 01 0 00 037747                      SKIPL   1,CRAMTO                ; 1st argument in range?
037732  303 01 0 00 007777                      CAILE   1,7777
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO     FIOFF
037733  254 00 0 00 057070                               RTN]
037734  274 01 0 00 037746                      SUB     1,CRAMFR                ; see if range ok
                                                JUMPL   1,[FMSGCD <? Range nonsensical>
                                                           GO   FIOFF
037735  321 01 0 00 057436                                 RTN]
```

```
                                ;#*********************************************************************
                                ; Process the command - examine, deposit, or zero
                                ;#*********************************************************************

                                ; Save default arguments
037736  200 00 0 00 037746              MOVE    CRAMFR                  ; set up the new default range
037737  202 00 0 00 037754              MOVEM   RFROM                   ;   selected, for later EXAM,
037740  200 00 0 00 037747              MOVE    CRAMTO                  ;   DEPOS, or ZEROs
037741  202 00 0 00 037753              MOVEM   RTO

                                ; Then ensure the port is stopped

037742  260 17 0 00 037656*             GO      ISTOP                   ; stop the port

                                ; Now do it

037743  476 00 0 00 064575'     DCOM1:  SETOM   DC1ST#                  ; init 'first location' flag
037744  200 01 0 00 064576'             MOVE    1,DFLAG                 ; get type of command
                                        JRST    @[DCOMEX                ; dispatch to proper routine
                                                  DCOMDE
                                                  DCOMDB
                                                  DCOMAL
                                                  DCOMZE
037745  254 00 1 01 060175                        DCOMLI](1)

                                ; Default CRAM locations, CRAM data, and Misc

037746  000000  000000          CRAMFR: 0                               ; range is from CRAMFR to
037747  000000  000000          CRAMTO: 0                               ;   CRAMTO
037750  000000  000000          CADDR:  0                               ; current address

037751  000000  000000          CWORDL: 0                               ; left 30 bits
037752  000000  000000          CWORDR: 0                               ; right 30 bits

037753  000000  000000          RTO:    0                               ; start address (default)
037754  000000  000000          RFROM:  0                               ; end address (default)

037755  000000  000000          ANEXT:  0                               ; next address
```

```

                                   ;#*********************************************************************
                                   ; Deposit into CRAM locations.
                                   ;#*********************************************************************

                                   DCOMDE: MOVEI   [SETZM  ALTF            ; set up altmode transfer address
                                                    GO     FIOFF           ;   to exit this command
037756  201 00 0 00 052541                          RTN]
037757  202 00 0 00 030063                 MOVEM   ALTMGO
037760  335 00 0 00 064575'                SKIPGE  DC1ST                   ; first location?
037761  037 01 0 00 060203                 FMSGC   <Type xxxx,xxxx,xxxx,xxxx,xxxx>
037762  402 00 0 00 064575'                SETZM   DC1ST                   ; clear 'first location' flag
037763  037 01 0 00 030242                 PCRLF
037764  476 00 0 00 037751                 SETOM   CWORDL                  ; initialize left 30 bits
037765  476 00 0 00 037752                 SETOM   CWORDR                  ; initialize right 30 bits
037766  200 00 0 00 037746                 MOVE    CRAMFR                  ; get the address to modify
037767  202 00 0 00 037750                 MOVEM   CADDR                   ; save address for writing
037770  037 04 0 00 000001                 PNT4F                           ; print it
037771  037 01 0 00 060212                 FMSG    <::    >
037772  403 04 0 00 000005                 SETZB   4,5                     ; clear data input locations
037773  476 00 0 00 034447*                SETOM   ARGFLG                  ; ensure that an argument is expected
037774  200 01 0 00 060223                 MOVE    1,[FMSGCD <xxxx,xxxx,xxxx,xxxx,xxxx (CR)>]
037775  260 17 0 00 037721*                GO      .OARG                   ; get first address
037776  263 17 0 00 000000                  RTN                            ; error/altmode/question - exit
                                            JRST   [FMSGCD <? Missing arguments>
                                                    GO     FIOFF
037777  254 00 0 00 060231                          RTN]
040000  200 00 0 00 037724*                MOVE    ARGUM                   ; get argument
040001  137 00 0 00 060234                 DPB     [POINT 12,4,11]         ; save it
040002  200 01 0 00 060223                 MOVE    1,[FMSGCD <xxxx,xxxx,xxxx,xxxx,xxxx (CR)>]
040003  260 17 0 00 037775*                GO      .OARG                   ; get first address
040004  263 17 0 00 000000                  RTN                            ; error/altmode/question - exit
                                            JRST   [FMSGCD <? Missing arguments>
                                                    GO     FIOFF
040005  254 00 0 00 060231                          RTN]
040006  200 00 0 00 040000*                MOVE    ARGUM                   ; get argument
040007  137 00 0 00 060235                 DPB     [POINT 12,4,23]         ; save it
040010  200 01 0 00 060223                 MOVE    1,[FMSGCD <xxxx,xxxx,xxxx,xxxx,xxxx (CR)>]
040011  260 17 0 00 040003*                GO      .OARG                   ; get first address
040012  263 17 0 00 000000                  RTN                            ; error/altmode/question - exit
                                            JRST   [FMSGCD <? Missing arguments>
                                                    GO     FIOFF
040013  254 00 0 00 060231                          RTN]
040014  200 00 0 00 040006*                MOVE    ARGUM                   ; get argument
040015  137 00 0 00 060236                 DPB     [POINT 12,4,35]         ; save it
```

```
040016  200 01 0 00 060223          MOVE    1,[FMSGCD <xxxx,xxxx,xxxx,xxxx,xxxx (CR)>]
040017  260 17 0 00 040011*         GO      .OARG                   ; get first address
040020  263 17 0 00 000000           RTN                            ; error/altmode/question - exit
                                      JRST  [FMSGCD <? Missing arguments>
                                              GO      FIOFF
040021  254 00 0 00 060231                    RTN]
040022  200 00 0 00 040014*         MOVE    ARGUM                   ; get argument
040023  137 00 0 00 060237          DPB     [POINT 12,5,11]         ; save it
040024  210 01 0 00 060223          MOVN    1,[FMSGCD <xxxx,xxxx,xxxx,xxxx,xxxx (CR)>]
040025  260 17 0 00 040017*         GO      .OARG                   ; get first address
040026  263 17 0 00 000000           RTN                            ; error/altmode/question - exit
                                      JRST  [FMSGCD <? Missing argument>
                                              GO      FIOFF
040027  254 00 0 00 055164                    RTN]
040030  200 00 0 00 040022*         MOVE    ARGUM                   ; get argument
040031  137 00 0 00 060240          DPB     [POINT 12,5,23]         ; save it
040032  135 00 0 00 060241          LDB     [POINT 30,4,29]         ; get left 30 bits
040033  202 00 0 00 037751          MOVEM   CWORDL                  ; save them
040034  246 04 0 00 777772          LSHC    4,-^D6                  ; position right 30 bits properly
040035  135 00 0 00 060242          LDB     [POINT 30,5,29]         ; get right 30 bits
040036  202 00 0 00 037752          MOVEM   CWORDR                  ; save them
040037  260 17 0 00 040376          GO      DWCRAM                  ; go write CRAM location
040040  350 01 0 00 037746          AOS     1,CRAMFR                ; update the specified address
040041  202 01 0 00 037755          MOVEM   1,ANEXT                 ; save next address
040042  317 01 0 00 037747          CAMG    1,CRAMTO                ; reach end yet?
040043  254 00 0 00 037756          JRST    DCOMDE                  ; no - loop till done
040044  263 17 0 00 000000          RTN                             ; yes - exit
```

```

                                        ;#**********************************************************************
                                        ; Deposit into CRAM locations (by bit field)
                                        ;#**********************************************************************

040045  402 00 0 00 037751              DCOMDB: SETZM   CWORDL                          ; initialize left 30 bits
040046  402 00 0 00 037752                      SETZM   CWORDR                          ; initialize right 30 bits
040047  200 00 0 00 037746                      MOVE    CRAMFR                          ; get the address to modify
040050  202 00 0 00 037750                      MOVEM   CADDR                           ; save address for writing
040051  254 00 0 00 040055                      JRST    DCOMAG

040052  200 00 0 00 037746              DCOMAL: MOVE    CRAMFR                          ; get the address to modify
040053  202 00 0 00 037750                      MOVEM   CADDR                           ; save address for writing
040054  260 17 0 00 040430                      GO      DRCRAM                          ; read CRAM (init both words)

                                        ; Set up initial data

040055  335 00 0 00 064575'             DCOMAG: SKIPGE  DC1ST                           ; first location?
                                                PNTMSF  [ASCIZ /
040056  037 01 0 00 060243              Type  value,value... or ^,CR,^Z,field name/]
040057  402 00 0 00 064575'                     SETZM   DC1ST                           ; clear 'first location' flag
040060  037 01 0 00 030242                      PCRLF
                                                MOVEI   [SETZM  ALTF                    ; set up altmode transfer address
                                                         GO     FIOFF                   ;   to exit this command
                                                         PCRLF
040061  201 00 0 00 060254                               RTN]
040062  202 00 0 00 030063                      MOVEM   ALTMGO
040063  200 00 0 00 037746                      MOVE    CRAMFR                          ; get the address to modify
040064  037 04 0 00 000001                      PNT4F                                   ; print it
040065  037 01 0 00 060260                      FMSGD   <::>
040066  260 17 0 00 034426*                     GO      CLRBUF                          ; clear input buffer
040067  402 00 0 00 030231                      SETZM   $TTCHR                          ; clear last character typed

                                        ; Select an entry from the bit table

040070  201 06 0 00 000001                      MOVEI   6,1                             ; init bit pointer
040071  275 06 0 00 000002              DCOMA0: SUBI    6,2                             ; decrement entry pointer
040072  315 06 0 00 060261                      CAMGE   6,[-1]                          ; backed up too far?
040073  474 06 0 00 000000                      SETO    6,                              ; force to start at beginning
040074  350 00 0 00 000006              DCOMA1: AOS     6                               ; point to next entry
040075  303 06 0 00 000021                      CAILE   6,^D17                          ; done with microword?
040076  254 00 0 00 040144                      JRST    DCOMAX                          ; yes - exit (write location)
```

```
                                        ; Have an entry - now get input/dispatch data

040077  200 00 0 00 030231                     MOVE    $TTCHR                  ; get last character typed
040100  302 00 0 00 000054                     CAIE    ","                     ; was it a comma
040101  306 00 0 00 000040                     CAIN    " "                     ;   or a blank?
040102  334 00 0 00 000000                     SKIPA                           ; yes - check file input
040103  254 00 0 00 040107                     JRST    DCOMAA                  ; no - continue
040104  332 00 0 00 064603'                    SKIPE   FINPUT                  ; file input?
                                               JRST    [GO     FINCMD          ; yes - input sixbit cmd
                                                        JRST   .+1             ; error or EOF
040105  254 00 0 00 060262                              JRST   DCOM3A]         ; continue
040106  254 00 0 00 040120                     JRST    DCOMA3
040107  332 00 0 00 064603'            DCOMAA: SKIPE   FINPUT                  ; file input?
                                               JRST    [GO     FINCMD          ; yes - input sixbit cmd
                                                        JRST   .+1             ; error or EOF
040110  254 00 0 00 060265                              JRST   DCOM3A]         ; continue
040111  200 00 0 00 064576'                    MOVE    DFLAG                   ; get dispatch flag
040112  306 00 0 00 000002                     CAIN    2                       ; alter?
040113  254 00 0 00 040117                     JRST    DCOMA2                  ; no - continue
040114  260 17 1 06 040177                     GO      @DCOLDB(6)              ; yes - get data byte
040115  037 17 0 00 000003                     PNTOCF
040116  037 01 0 00 000040                     PSPF

040117  256 00 0 06 040155             DCOMA2: XCT     DCOMSG(6)               ; no - print field name
040120  037 10 0 00 000003             DCOMA3: TTSIXB                          ; get input
                                               ALTCHK  [GET     XXW#           ; adjust stack properly
                                                        SKIPN   $TWCNT         ; did a timeout occur?
                                                        JRST    DCOMA3         ; yes - keep looking for input
                                                        MOVE    $TTCHR         ; get character typed
                                                        CAIN    "^"            ; backup character?
                                                        JRST    DCOMA0         ; yes - backup
                                                        CAIN    32             ; control-Z?
                                                        PCRLF                  ; yes - a final blank line
                                                        CAIN    32             ; control-Z?
                                                        JRST    DCOMAX         ; yes - exit - done
                                                        CAIN    "?"            ; question mark?
                                                        JRST    [XCT    MWDQUE(6); yes - handle
                                                                 PCRLF
                                                                 JRST   DCOMAA]
                                                        FMSGCD <? Argument error>
                                                        GO      CLRBUF
                                                        PCRLF
                                                        GO      FIOFF
                                                        JRST    DCOMA2]        ; no - error - reask question
040121  007 00 0 00 060300             DCOM3A: JUMPE   [MOVE    $TTCHR         ; anything typed?  if a space, keep
                                                        CAIN    40             ;   looking for an argument, if not
                                                        JRST    DCOMA3         ;   exit
040122  322 00 0 00 060321                              JRST    DCOMA5]
040123  202 00 0 00 064636'                    MOVEM   SAVFLD#                 ; save argument
040124  200 01 0 00 064636'                    MOVE    1,SAVFLD                ; get into AC1
040125  260 17 0 00 034331*                    GO      CONVSX                  ; convert to octal
040126  334 00 0 00 000000                     SKIPA
040127  254 00 0 00 040137                     JRST    DCOMA4                  ; ok - insert into data
```

```

                                              ; Unrecognizable - check if a field was specified

 040130  200 01 0 00 064636'                         MOVE    1,SAVFLD                ; get command into AC1
 040131  201 02 0 00 040243                          MOVEI   2,FLDLIS                ; get address of field list
 040132  260 17 0 00 035411*                         GO      .COMM                   ; handle command decoding
                                                      JRST   [FMSGCD <? Argument error>
                                                              GO      CLRBUF
                                                              PCRLF
                                                              GO      FIOFF
 040133  254 00 0 00 060314                                   JRST    DCOMA2]        ; no - error - reask question
 040134  370 00 0 00 000001                          SOS     1                       ; normalize to -1..12
 040135  200 06 0 00 000001                          MOVE    6,1                     ; put into AC6
 040136  254 00 0 00 040074                          JRST    DCOMA1                  ; go handle the field

                                              ; Insert field entry

 040137  200 00 0 00 000001                   DCOMA4: MOVE    1                       ; get number
 040140  260 17 1 06 040221                           GO      @DCODPB(6)              ; yes - insert it
 040141  200 00 0 00 030231                   DCOMA5: MOVE    $TTCHR                  ; get last character typed
 040142  312 00 0 00 000015                           CAME    15                      ; CRLF?
 040143  254 00 0 00 040074                           JRST    DCOMA1                  ; no - get next entry

                                              ; Finally, write the location

 040144  260 17 0 00 040376                   DCOMAX: GO      DWCRAM                  ; go write CRAM location
 040145  350 01 0 00 037746                           AOS     1,CRAMFR                ; update the specified address
 040146  202 01 0 00 037755                           MOVEM   1,ANEXT                 ; save next address
 040147  313 01 0 00 037747                           CAMLE   1,CRAMTO                ; reach end yet?
 040150  263 17 0 00 000000                           RTN                             ; yes - exit
 040151  200 00 0 00 064576'                          MOVE    DFLAG                   ; get dispatch flag
 040152  302 00 0 00 000003                           CAIE    3                       ; alter?
 040153  254 00 0 00 040045                           JRST    DCOMDB                  ; no - deposit by bit field
 040154  254 00 0 00 040052                           JRST    DCOMAL                  ; yes - alter by bit field
```

```


                                        ; Input message table

040155  037 01 0 00 060325              DCOMSG: FMSG    <J - >
040156  037 01 0 00 060326                      FMSG    <PAR - >
040157  037 01 0 00 060330                      FMSG    <OENA - >
040160  037 01 0 00 060332                      FMSG    <MGC - >
040161  037 01 0 00 060334                      FMSG    <SORC - >
040162  037 01 0 00 060336                      FMSG    <FUNC - >
040163  037 01 0 00 060340                      FMSG    <DEST - >
040164  037 01 0 00 060342                      FMSG    <CENA - >
040165  037 01 0 00 060344                      FMSG    <RAM - >
040166  037 01 0 00 060346                      FMSG    <A - >
040167  037 01 0 00 060347                      FMSG    <B - >
040170  037 01 0 00 060350                      FMSG    <SK - >
040171  037 01 0 00 060352                      FMSG    <BUS - >
040172  037 01 0 00 060354                      FMSG    <CRY - >
040173  037 01 0 00 060356                      FMSG    <CTL - >
040174  037 01 0 00 060360                      FMSG    <TIME - >
040175  037 01 0 00 060362                      FMSG    <SPARE - >
040176  037 01 0 00 060364                      FMSG    <MARK - >

                                        ; Microword load byte pointer table


040177  000000  060367                  DCOLDB: [LDB    [POINT 12,CWORDL,17]    ; J - Bits 6-17
                                                  RTN]
040200  000000  060372                          [LDB    [POINT 1,CWORDL,18]     ; PAR - Bit 18
                                                  RTN]
040201  000000  060375                          [LDB    [POINT 1,CWORDL,19]     ; OENA - Bit 19
                                                  RTN]
040202  000000  060400                          [LDB    [POINT 10,CWORDL,29]    ; MGC - Bits 20-29
                                                  RTN]
040203  000000  060403                          [LDB    [POINT 3,CWORDL,32]     ; SORC - Bits 30-32
                                                  RTN]
040204  000000  060406                          [LDB    [POINT 3,CWORDL,35]     ; FUNC - Bits 33-35
                                                  RTN]
040205  000000  060411                          [LDB    [POINT 3,CWORDR,8]      ; DEST - Bits 6-8
                                                  RTN]
040206  000000  060414                          [LDB    [POINT 1,CWORDR,9]      ; CENA - Bit 9
                                                  RTN]
040207  000000  060417                          [LDB    [POINT 1,CWORDR,10]     ; RAM - Bit 10
                                                  RTN]
040210  000000  060422                          [LDB    [POINT 4,CWORDR,14]     ; A - Bits 11-14
                                                  RTN]
040211  000000  060425                          [LDB    [POINT 4,CWORDR,18]     ; B - Bits 15-18
                                                  RTN]
040212  000000  060430                          [LDB    [POINT 5,CWORDR,23]     ; SK - Bits 19-23
                                                  RTN]
040213  000000  060433                          [LDB    [POINT 3,CWORDR,26]     ; BUS - Bits 24-26
                                                  RTN]
```

```
4751                                      [LDB    [POINT 1,CWORDR,27]     ; CRY - Bits 27
4752  040214  000000  060436               RTN]
4753                                      [LDB    [POINT 4,CWORDR,31]     ; OP - Bits 28-31
4754  040215  000000  060441               RTN]
4755                                      [LDB    [POINT 1,CWORDR,32]     ; TIME - Bit 32
4756  040216  000000  060444               RTN]
4757                                      [LDB    [POINT 2,CWORDR,34]     ; SPARE - Bits 33-34
4758  040217  000000  060447               RTN]
4759                                      [LDB    [POINT 1,CWORDR,35]     ; MARK - Bit 35
4760  040220  000000  060452               RTN]
4761
4762                             ; Microword deposit byte pointer table
4763
4764                             DCODPB: [DPB    [POINT 12,CWORDL,17]    ; J - Bits 6-17
4765  040221  000000  060454               RTN]
4766                                      [DPB    [POINT 1,CWORDL,18]     ; PAR - Bit 18
4767  040222  000000  060456               RTN]
4768                                      [DPB    [POINT 1,CWORDL,19]     ; OENA - Bit 19
4769  040223  000000  060460               RTN]
4770                                      [DPB    [POINT 10,CWORDL,29]    ; MGC - Bits 20-29
4771  040224  000000  060462               RTN]
4772                                      [DPB    [POINT 3,CWORDL,32]     ; SORC - Bits 30-32
4773  040225  000000  060464               RTN]
4774                                      [DPB    [POINT 3,CWORDL,35]     ; FUNC - Bits 33-35
4775  040226  000000  060466               RTN]
4776                                      [DPB    [POINT 3,CWORDR,8]      ; DEST - Bits 6-8
4777  040227  000000  060470               RTN]
4778                                      [DPB    [POINT 1,CWORDR,9]      ; CENA - Bit 9
4779  040230  000000  060472               RTN]
4780                                      [DPB    [POINT 1,CWORDR,10]     ; RAM - Bit 10
4781  040231  000000  060474               RTN]
4782                                      [DPB    [POINT 4,CWORDR,14]     ; A - Bits 11-14
4783  040232  000000  060476               RTN]
4784                                      [DPB    [POINT 4,CWORDR,18]     ; B - Bits 15-18
4785  040233  000000  060500               RTN]
4786                                      [DPB    [POINT 5,CWORDR,23]     ; SK - Bits 19-23
4787  040234  000000  060502               RTN]
4788                                      [DPB    [POINT 3,CWORDR,26]     ; BUS - Bits 24-26
4789  040235  000000  060504               RTN]
4790                                      [DPB    [POINT 1,CWORDR,27]     ; CRY - Bits 27
4791  040236  000000  060506               RTN]
4792                                      [DPB    [POINT 4,CWORDR,31]     ; OP - Bits 28-31
4793  040237  000000  060510               RTN]
4794                                      [DPB    [POINT 1,CWORDR,32]     ; TIME - Bit 32
4795  040240  000000  060512               RTN]
4796                                      [DPB    [POINT 2,CWORDR,34]     ; SPARE - Bits 33-34
4797  040241  000000  060514               RTN]
4798                                      [DPB    [POINT 1,CWORDR,35]     ; MARK - Bit 35
4799  040242  000000  060516               RTN]
```

```
 4800
 4801                                         ; Field command table
 4802
 4803  040243  52 00 00 00 00 00              FLDLIS: SIXBIT  /J/
 4804  040244  60 41 62 00 00 00                      SIXBIT  /PAR/
 4805  040245  57 45 56 41 00 00                      SIXBIT  /OENA/
 4806  040246  55 47 43 00 00 00                      SIXBIT  /MGC/
 4807  040247  63 57 62 43 00 00                      SIXBIT  /SORC/
 4808  040250  46 65 56 43 00 00                      SIXBIT  /FUNC/
 4809  040251  44 45 63 64 00 00                      SIXBIT  /DEST/
 4810  040252  43 45 56 41 00 00                      SIXBIT  /CENA/
 4811  040253  62 41 55 00 00 00                      SIXBIT  /RAM/
 4812  040254  41 00 00 00 00 00                      SIXBIT  /A/
 4813  040255  42 00 00 00 00 00                      SIXBIT  /B/
 4814  040256  63 53 00 00 00 00                      SIXBIT  /SK/
 4815  040257  42 65 63 00 00 00                      SIXBIT  /BUS/
 4816  040260  43 62 71 00 00 00                      SIXBIT  /CRY/
 4817  040261  43 64 54 00 00 00                      SIXBIT  /CTL/
 4818  040262  64 51 55 45 00 00                      SIXBIT  /TIME/
 4819  040263  63 60 41 62 45 00                      SIXBIT  /SPARE/
 4820  040264  55 41 62 53 00 00                      SIXBIT  /MARK/
 4821  040265  000000  000000                         0
```

```
                                         ; Help text

040266  037 01 0 00 060520       MWDQUE: FMSGCD  <Jump field>
040267  037 01 0 00 060523               FMSGCD  <Parity bit - Pgm calculates it unless SET PAR is off>
040270  037 01 0 00 060537               FMSGCD  <Ouput enable>
040271  037 01 0 00 060543               FMSGCD  <Magic # field>
                                         FMSGD   <
                                 Source field:
                                  0-AQ  2-0Q  4-0A  6-DQ
040272  037 01 0 00 060547        1-AB  3-0B  5-DA  7-D0>
                                         FMSGD   <
                                 ALU Function:
                                  0-Plus  2-Rmin  4-And   6-Xor
040273  037 01 0 00 060565        1-Smin  3-Or    5-Nand  7-Xnor>
                                         FMSGD   <
                                 Destination field:
                                  0-QF   2-BA  4-/2BQF  6-*2BQF
040274  037 01 0 00 060606        1-F    3-BF  5-/2BF   7-*2BF>
040275  037 01 0 00 060627               FMSGCD  <CC Enable>
040276  037 01 0 00 060632               FMSGCD  <RAM Mode>
040277  037 01 0 00 060635               FMSGCD  <Port A 0-17>
040300  037 01 0 00 060641               FMSGCD  <Port B 0-17>
                                         FMSGD   <
                                 Skip field - Select:
                                  00/20-CCCbusAvail  10/30-CCEbusRqst
                                  01/21-CCGrntCsr    11/31-CCOverflow
                                  02/22-CCFeq0       12/32-CCMbSign
                                  03/23-CCCsrChng    13/33-CCMVParChk
                                  04/24-CCEbParerr   14/34-CCCbusParErr
                                  05/25-CCRcvBufAFul 15/35-CCPliParErr
                                  06/26-CCRcvBufBFul 16/36-CCChanErr
                                  07/27-CCXmtrAttn   17/37-CCCbLstWd

                                  20/30-LoadSadReg   23/33-LdLocalMem
                                  21/31-SelMbusFld   24/34-SelCnstFld
040301  037 01 0 00 060645        22/32-RdLocalMem>
                                         FMSGD   <
                                 Bus Ctl field - Select:
                                  0-Nul  2-Mbus  4-Cbus
040302  037 01 0 00 060772        1-Pli  3-Fmtr  5-Ebus>
040303  037 01 0 00 061012               FMSGCD  <Carry input>
                                         FMSGD   <
                                 2910 Control field:
                                  00-JZ    04-PUSH  10-RFCT  14-LDCT
                                  01-CJS   05-JSRP  11-RPCT  15-LOOP
                                  02-JMAP  06-CJV   12-CRTN  16-CONT
040304  037 01 0 00 061016        03-CJP   07-JRP   13-CJPP  17-TWB>
040305  037 01 0 00 061061               FMSGCD  <Time field>
040306  037 01 0 00 061064               FMSGCD  <Spare bits>
040307  037 01 0 00 061067               FMSGCD  <Mark bit>
```

```
                                            ;#**********************************************************************
                                            ; Zero CRAM addresses specified by 'PARG1'' and 'PARG2''
                                            ;#**********************************************************************

040310  402 00 0 00 037751          DCOMZE: SETZM   CWORDL                  ; clear CRAM words left and
                                            SETZM   CWORDR                  ;   right
040311  402 00 0 00 037752
040312  200 00 0 00 037746                  MOVE    CRAMFR                  ; get the address to modify
040313  202 00 0 00 037750                  MOVEM   CADDR                   ; save address for writing
040314  260 17 0 00 040376                  GO      DWCRAM                  ; go write CRAM location
040315  037 07 0 00 000003                  TTALTM                          ; altmode key struck?
040316  334 00 0 00 000000                  SKIPA                           ; no - proceed
040317  263 17 0 00 000000                  RTN                             ; yes - exit
040320  350 01 0 00 037746                  AOS     1,CRAMFR                ; update the specified address
040321  202 01 0 00 037755                  MOVEM   1,ANEXT                 ; save next address
040322  317 01 0 00 037747                  CAMG    1,CRAMTO                ; reach end yet?
040323  254 00 0 00 040310                  JRST    DCOMZE                  ; no - loop till done
040324  263 17 0 00 000000                  RTN                             ; yes - exit



                                            ;#**********************************************************************
                                            ; Examine CRAM locations
                                            ;#**********************************************************************

040325  200 00 0 00 037746          DCOMEX: MOVE    CRAMFR                  ; get the address to modify
040326  202 00 0 00 037750                  MOVEM   CADDR                   ; save address
040327  037 00 0 00 030242                  PCRL
040330  001 04 0 00 037750                  PNTOCC  4,CADDR                 ; print address
040331  037 00 0 00 061072                  TMSG    </ >
040332  260 17 0 00 040430                  GO      DRCRAM                  ; read CRAM location
040333  255 00 0 00 000000                  JFCL                            ; error return
040334  200 02 0 00 037751                  MOVE    2,CWORDL                ; get left half
040335  200 03 0 00 037752                  MOVE    3,CWORDR                ; get right half
040336  260 17 0 00 043603                  GO      PNTWD                   ; print it
040337  037 07 0 00 000003                  TTALTM                          ; altmode key struck?
040340  334 00 0 00 000000                  SKIPA                           ; no - proceed
                                            JRST    [SETOM  ALTF            ; yes - set altmode typed flag
                                                     SETZM  MULFLG          ;   and clear 'examine next' flag
040341  254 00 0 00 061073                           JRST   DCOMEZ]         ;   and exit
040342  350 01 0 00 037746                  AOS     1,CRAMFR                ; update the specified address
040343  202 01 0 00 037755                  MOVEM   1,ANEXT                 ; save next address
040344  317 01 0 00 037747                  CAMG    1,CRAMTO                ; reach end yet?
040345  254 00 0 00 040325                  JRST    DCOMEX                  ; no - loop till done
040346  336 00 0 00 064616'         DCOMEZ: SKIPN   MULFLG                  ; in 'examine next'?
040347  037 00 0 00 030242                  PCRL                            ; no - print blank line
040350  263 17 0 00 000000                  RTN                             ; yes - exit
```

```
                                       ;#************************************************************************
                                       ; List CRAM locations
                                       ;#************************************************************************

                                       ; Print this CRAM entry
040351  332 00 0 00 064617'            DCOMLI: SKIPE   MULINI                   ; in 'examine next'?
040352  037 00 0 00 061076                     TMSGC   <ADDR/ JUMP PAR OUT MGC SOR FUN DES CC RAM PA PB SK BUS CRY CTL  TM SP MK>
040353  200 00 0 00 037746             DCOMLO: MOVE    CRAMFR                   ; get the address to list
040354  202 00 0 00 037750                     MOVEM   CADDR                    ; save address
040355  260 17 0 00 040430                     GO      DRCRAM                   ; read CRAM location
040356  255 00 0 00 000000                     JFCL                             ; error return
040357  037 00 0 00 030242                     PCRL                             ; start a new line
040360  001 04 0 00 037750                     PNTOCC  4,CADDR                  ; print cram address
040361  037 00 0 00 061072                     TMSG    </ >
040362  200 02 0 00 037751                     MOVE    2,CWORDL                 ; get left word
040363  200 03 0 00 037752                     MOVE    3,CWORDR                 ; get right word
040364  260 17 0 00 043507                     GO      PNTCRM                   ; go print it
040365  332 00 0 00 064570'                    SKIPE   ALTF                     ; altmode typed?
040366  254 00 0 00 040373                     JRST    DCOMLX                   ; yes - return
040367  350 01 0 00 037746                     AOS     1,CRAMFR                 ; no - update the specified address
040370  202 01 0 00 037755                     MOVEM   1,ANEXT                  ; save next address
040371  317 01 0 00 037747                     CAMG    1,CRAMTO                 ; reach end yet?
040372  254 00 0 00 040353                     JRST    DCOMLO                   ; no - loop till done
040373  336 00 0 00 064616'            DCOMLX: SKIPN   MULFLG                   ; in 'examine next'?
040374  037 00 0 00 030242                     PCRL                             ; no - print blank line
040375  263 17 0 00 000000                     RTN                              ; yes - exit
```

```
                                              ;#**********************************************************************
                                              ; DWCRAM - Load CRAM word at 'CADDR' with data in 'CWORDL' and 'CWORDR'
                                              ;#**********************************************************************

040376  261 17 0 00 000000                    DWCRAM: RPUT    (0,1,2,3)               ; save AC's
040402  200 01 0 00 037750                            MOVE    1,CADDR                 ; get actual CRAM load address
040403  242 01 0 00 000001                            LSH     1,1                     ; convert to address for RAR
040404  660 01 0 00 000001                            TRO     1,1                     ; set bit 12 (for left 30 bits)
040405  260 17 0 00 037661*                           GO      LDRAR                   ; load RAR with it
040406  200 02 0 00 037751                            MOVE    2,CWORDL                ; get left 30 bits
040407  200 03 0 00 037752                            MOVE    3,CWORDR                ; get right 30 bits
040410  400 01 0 00 000000                            SETZ    1,                      ; ignore 'bad parity' bit
040411  332 00 0 00 035764                            SKIPE   PARFLG                  ; calculate parity?
040412  260 17 0 00 043625                            GO      CALPAR                  ; yes - do so
040413  200 01 0 00 000002                            MOVE    1,2                     ; get left 30 bits
040414  260 17 0 00 000000*                           GO      LDCRAM                  ; load it

040415  200 01 0 00 037750                            MOVE    1,CADDR                 ; get actual CRAM load address
040416  242 01 0 00 000001                            LSH     1,1                     ; convert to address for RAR
040417  620 01 0 00 000001                            TRZ     1,1                     ; clear bit 12 (for right 30 bits)
040420  260 17 0 00 040405*                           GO      LDRAR                   ; load address
040421  200 01 0 00 000003                            MOVE    1,3                     ; get right 30 bits
040422  260 17 0 00 040414*                           GO      LDCRAM                  ; load it
040423  262 17 0 00 000003                            RGET    (3,2,1,0)               ; restore AC's
040427  263 17 0 00 000000                            RTN

                                              ;#**********************************************************************
                                              ; DRCRAM - Read CRAM word at 'CADDR' into 'CWORDL' and 'CWORDR'
                                              ;#**********************************************************************

040430  261 17 0 00 000000                    DRCRAM: RPUT    (0,1)                   ; save AC's
040432  200 01 0 00 037750                            MOVE    1,CADDR                 ; get actual CRAM address
040433  242 01 0 00 000001                            LSH     1,1                     ; convert to address for RAR
040434  660 01 0 00 000001                            TRO     1,1                     ; set bit 12 (for left 30 bits)
040435  260 17 0 00 040420*                           GO      LDRAR                   ; load RAR with it
040436  260 17 0 00 000000*                           GO      RDCRAM                  ; load it
040437  202 01 0 00 037751                            MOVEM   1,CWORDL                ; save it

040440  200 01 0 00 037750                            MOVE    1,CADDR                 ; get actual CRAM load address
040441  242 01 0 00 000001                            LSH     1,1                     ; convert to address for RAR
040442  620 01 0 00 000001                            TRZ     1,1                     ; clear bit 12 (for right 30 bits)
040443  260 17 0 00 040435*                           GO      LDRAR                   ; load address
040444  260 17 0 00 040436*                           GO      RDCRAM                  ; load it
040445  202 01 0 00 037752                            MOVEM   1,CWORDR                ; save it
040446  262 17 0 00 000001                            RGET    (1,0)                   ; restore AC's
040450  263 17 0 00 000000                            RTN                             ; return
```

```
                                 ;#*********************************************************************
                                 ;* DEALU - Examine 2901 registers
                                 ;
                                 ; Arguments:  Reg1,Reg2
                                 ;#*********************************************************************

040451  200 00 0 00 040753       DEALU:  MOVE    ALUFR                      ; set up default of register range
040452  202 00 0 00 064626'              MOVEM   RALUB#                     ;   to be whatever was typed before
040453  200 00 0 00 040754               MOVE    ALUTO                      ;   or if nothing typed, then 0-20
040454  202 00 0 00 064627'              MOVEM   RALUE#

                                 ; Get arguments

040455  200 01 0 00 061124               MOVE    1,[FMSGCD <E2901 reg1,reg2 (CR) (20=QReg)>]
040456  260 17 0 00 040025*              GO      .OARG                      ; get 1st argument
040457  263 17 0 00 000000                RTN                               ; error/altmode/question - exit
040460  254 00 0 00 040472                 JRST  DEALU0                     ; no arguments given - continue
040461  200 00 0 00 040030*              MOVE    ARGUM                      ; get argument
040462  202 00 0 00 064626'              MOVEM   RALUB                      ; save in 'from' location
040463  202 00 0 00 064627'              MOVEM   RALUE                      ; save in 'to' location

040464  210 01 0 00 061132               MOVN    1,[FMSGCD <E2901 reg1,reg2 (CR)>]
040465  260 17 0 00 040456*              GO      .OARG                      ; get 2nd argument
040466  263 17 0 00 000000                RTN                               ; error/altmode/question - exit
040467  254 00 0 00 040472                 JRST  DEALU0                     ; no arguments given - continue
040470  200 00 0 00 040461*              MOVE    ARGUM                      ; get argument
040471  202 00 0 00 064627'              MOVEM   RALUE                      ; save in 'to' location

                                 ; Now range check the arguments

040472  331 01 0 00 064626'      DEALU0: SKIPL   1,RALUB                    ; 1st argument in range?
040473  303 01 0 00 000020               CAILE   1,20
                                         JRST    [FMSGCD <? Range of 2901 registers is 0-20 (20=QReg)>
                                                   SETOM  ALTF
                                                  GO      FIOFF
040474  254 00 0 00 061145                        RTN]
040475  331 01 0 00 064627'              SKIPL   1,RALUE                    ; 2nd argument in range?
040476  303 01 0 00 000020               CAILE   1,20
                                         JRST    [FMSGCD <? Range of 2901 registers is 0-20 (20=QReg)>
                                                  GO      FIOFF
040477  254 00 0 00 061151                        RTN]
040500  274 01 0 00 064626'              SUB     1,RALUB                    ; see if range ok
                                         JUMPL   1,[FMSGCD <? Range nonsensical>
                                                  GO      FIOFF
040501  321 01 0 00 057436                        RTN]
040502  200 00 0 00 064626'              MOVE    RALUB                      ; get 'from' argument
040503  202 00 0 00 040753               MOVEM   ALUFR                      ; save it
040504  200 00 0 00 064627'              MOVE    RALUE                      ; get 'to' argument
040505  202 00 0 00 040754               MOVEM   ALUTO                      ; save it
040506  201 00 0 00 000006               MOVEI   6                          ; get E2901 flag
040507  202 00 0 00 064576'              MOVEM   DFLAG                      ; save it
```

```



                                        ; Save CRAM location

040510  200 01 0 00 061154                       MOVE    1,[-1,,7750]              ; set up AOBJN word
040511  260 17 0 00 044040                       GO      SAVCRM                    ; save the CRAM locations

                                        ; Now do each register

040512  037 00 0 00 061155              DEALU1:  TMSGC   <R>                       ; start a new line
040513  200 02 0 00 064626'                      MOVE    2,RALUB                   ; get register number
040514  200 00 0 00 000002                       MOVE    2                         ; get register number
040515  137 02 0 00 061156                       DPB     2,[POINT 4,DEALUM+1,14]   ; save it in microword
040516  306 02 0 00 000020                       CAIN    2,20                      ; Q-Register?
040517  037 00 0 00 061157                       TMSG    <Q>                       ; yes - print Q
040520  302 02 0 00 000020                       CAIE    2,20                      ; Q-Register?
040521  037 16 0 00 000003                       PNTOCS                            ; no - print register number
040522  037 00 0 00 061072                       TMSG    </ >
040523  201 00 0 00 000004                       MOVEI   4                         ; get source field of 4 (OA)
040524  306 02 0 00 000020                       CAIN    2,20                      ; Q-Register?
040525  201 00 0 00 000002                       MOVEI   2                         ; yes - use source of 2 (OQ)
040526  137 00 0 00 061160                       DPB     [POINT 3,DEALUM+1,2]      ; set up source field
040527  201 01 0 00 040732                       MOVEI   1,DEALUM                  ; set up microcode address
040530  260 17 0 00 043207                       GO      MLOADN                    ; load/verify it
040531  255 00 0 00 000000                       JFCL                              ; ignore error
040532  201 01 0 00 007750                       MOVEI   1,7750                    ; get start address
040533  242 01 0 00 000001                       LSH     1,1                       ; position correctly
040534  260 17 0 00 040443*                      GO      LDRAR                     ; load the RAR
040535  201 01 0 00 220010                       MOVEI   1,SINCYC!MPRUN!TSTEBF     ; single step once
040536  260 17 0 00 037660*                      GO      LDCSR
040537  260 17 0 00 037574*                      GO      RDEBUF                    ; read EBUF contents
040540  200 00 0 00 000001                       MOVE    1                         ; get register data
040541  037 13 0 00 000000                       PNTHW                             ; print it
040542  350 00 0 00 000002                       AOS     2                         ; point to next register
040543  202 02 0 00 037755                       MOVEM   2,ANEXT                   ; save next register
040544  037 07 0 00 000003                       TTALTM                            ; altmode key struck?
040545  334 00 0 00 000000                       SKIPA                             ; no - proceed
                                                 JRST    [SETOM  ALTF              ; yes - set altmode typed flag
                                                          SETZM  MULFLG            ;   and clear 'examine next' flag
                                                          JRST   DEALUX]           ;   and exit
040546  254 00 0 00 061161
040547  350 01 0 00 064626'                      AOS     1,RALUB                   ; point to next register
040550  317 01 0 00 064627'                      CAMG    1,RALUE                   ; done yet?
040551  254 00 0 00 040512                       JRST    DEALU1                    ; no - loop till done

                                        ; Restore CRAM location and exit

040552  336 00 0 00 064616'             DEALUX:  SKIPN   MULFLG                    ; in 'examine next'?
040553  037 00 0 00 030242                       PCRL                              ; no - print blank line
040554  200 01 0 00 061154                       MOVE    1,[-1,,7750]              ; set up AOBJN word
040555  260 17 0 00 044062                       GO      RESCRM                    ; restore CRAM data
040556  263 17 0 00 000000                       RTN                               ; return
```

```
                                        ;#**********************************************************************
                                        ;* DDALU - Deposit register
                                        ;
                                        ; Format:  DALU reg,data
                                        ;#**********************************************************************

040557  200 01 0 00 061171              DDALU:  MOVE    1,[FMSGCD <DALU reg,data (CR)>]
040560  260 17 0 00 036255*                     GO      .DARG                   ; get register number
040561  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
                                                 JRST   [FMSGCD <? Missing argument>
                                                         GO     FIOFF
040562  254 00 0 00 055164                               RTN]
040563  331 01 0 00 040470*             DDALU0: SKIPL   1,ARGUM                 ; argument in range?
040564  303 01 0 00 000020                      CAILE   1,20
                                                JRST    [FMSGCD <? Range of 2901 registers is 0-20 (20=QReg)>
                                                         GO     FIOFF
040565  254 00 0 00 061151                               RTN]
040566  202 01 0 00 064626'                     MOVEM   1,RALUB                 ; get 'from' argument
040567  202 01 0 00 040753                      MOVEM   1,ALUFR                 ; save it
040570  202 01 0 00 064627'                     MOVEM   1,RALUE                 ; get 'to' argument
040571  202 01 0 00 040754                      MOVEM   1,ALUTO                 ; save it
040572  200 01 0 00 061171                      MOVE    1,[FMSGCD <DALU reg,data (CR)>]
040573  260 17 0 00 040465*                     GO      .OARG                   ; get data
040574  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
                                                 JRST   [FMSGCD <? Missing argument>
                                                         GO     FIOFF
040575  254 00 0 00 055164                               RTN]
040576  200 00 0 00 040563*                     MOVE    ARGUM                   ; get argument
040577  202 00 0 00 064607'                     MOVEM   LARG2                   ; save it
040600  201 00 0 00 000007                      MOVEI   7                       ; get D2901 flag
040601  202 00 0 00 064576'                     MOVEM   DFLAG                   ; save it

                                        ; Save CRAM locations

040602  200 01 0 00 061172                      MOVE    1,[-5,,7750]            ; set up AOBJN word
040603  260 17 0 00 044040                      GO      SAVCRM                  ; save the CRAM locations

                                        ; Load microcode

040604  200 02 0 00 064626'                     MOVE    2,RALUB                 ; get location to write
040605  137 02 0 00 061173                      DPB     2,[POINT 4,DDALUM+7,18] ; set up 2901 register
040606  201 00 0 00 000002                      MOVEI   2                       ; get destination field of 2 (BA)
040607  306 02 0 00 000020                      CAIN    2,20                    ; Q-Register?
040610  201 00 0 00 000000                      MOVEI   0                       ; yes - use destination of 0 (QF)
040611  137 00 0 00 061174                      DPB     [POINT 3,DDALUM+7,8]    ; set up source field
040612  201 01 0 00 040735                      MOVEI   1,DDALUM                ; set up microcode address
040613  260 17 0 00 043207                      GO      MLOADN                  ; load/verify it
040614  255 00 0 00 000000                      JFCL                            ; ignore error
040615  350 00 0 00 000002                      AOS     2                       ; point to next register
040616  202 02 0 00 037755                      MOVEM   2,ANEXT                 ; save next register
```

```
                                                  ; Now do the deposit

040617  201 01 0 00 007750                                MOVEI   1,7750                  ; get start address
040620  242 01 0 00 000001                                LSH     1,1                     ; position correctly
040621  260 17 0 00 040534*                               GO      LDRAR                   ; load the RAR
040622  201 01 0 00 000010                                MOVEI   1,MPRUN                 ; set initial data
040623  260 17 0 00 040536*                               GO      LDCSR                   ; start the port
040624  200 01 0 00 064607'                               MOVE    1,LARG2                 ; get data to write
040625  260 17 0 00 000000*                               GO      .DATAO                  ; give to port
040626  400 01 0 00 000000                                SETZ    1,                      ; stop the port
040627  260 17 0 00 040623*                               GO      LDCSR

                                                  ; Check if additional arguments

040630  336 00 0 00 037773*                               SKIPN   ARGFLG                  ; any more arguments
040631  254 00 0 00 040635                                JRST    DDALUX                  ; no - exit
040632  200 00 0 00 037755                                MOVE    ANEXT                   ; get register number
040633  202 00 0 00 040576*                               MOVEM   ARGUM                   ; save it
040634  254 00 0 00 040563                                JRST    DDALU0                  ; loop till done

                                                  ; Restore CRAM locations and exit

040635  200 01 0 00 061172                        DDALUX: MOVE    1,[-5,,7750]            ; set up AOBJN word
040636  260 17 0 00 044062                                GO      RESCRM                  ; restore CRAM data
040637  263 17 0 00 000000                                RTN                             ; return
```

```
                                     ;#**********************************************************************
                                     ;* DZALU - Zero 2901 registers
                                     ;
                                     ; Format:  ZALU reg1,reg2
                                     ;#**********************************************************************

040640  200 00 0 00 040753           DZALU:  MOVE    ALUFR                     ; set up default of register range
040641  202 00 0 00 064626'                  MOVEM   RALUB                     ;   to be whatever was typed before
040642  200 00 0 00 040754                   MOVE    ALUTO                     ;   or if nothing typed, then 0-0
040643  202 00 0 00 064627'                  MOVEM   RALUE

                                     ; Get arguments

040644  200 01 0 00 061202                   MOVE    1,[FMSGCD <ZALU reg1,reg2 (CR)>]
040645  260 17 0 00 040560*                  GO      .DARG                     ; get first argument
040646  263 17 0 00 000000                    RTN                              ; error/altmode/question - exit
040647  254 00 0 00 040661                     JRST  DZALU0                    ; no arguments given - continue
040650  200 00 0 00 040633*                  MOVE    ARGUM                     ; get argument
040651  202 00 0 00 064626'                  MOVEM   RALUB                     ; save in 'from' location
040652  202 00 0 00 064627'                  MOVEM   RALUE                     ; save in 'to' location

040653  210 01 0 00 061202                   MOVN    1,[FMSGCD <ZALU reg1,reg2 (CR)>]
040654  260 17 0 00 040645*                  GO      .DARG                     ; get 2nd argument
040655  263 17 0 00 000000                    RTN                              ; error/altmode/question - exit
040656  254 00 0 00 040661                     JRST  DZALU0                    ; no arguments given - continue
040657  200 00 0 00 040650*                  MOVE    ARGUM                     ; get argument
040660  202 00 0 00 064627'                  MOVEM   RALUE                     ; save in 'to' location

                                     ; Now range check the arguments

040661  331 01 0 00 064626'          DZALU0: SKIPL   1,RALUB                   ; 1st argument in range?
040662  303 01 0 00 000020                   CAILE   1,20
                                             JRST    [FMSGCD <? Range of 2901 registers is 0-20 (20=QReg)>
                                                      GO     FIOFF
040663  254 00 0 00 061151                            RTN]
040664  331 01 0 00 064627'                  SKIPL   1,RALUE                   ; 2nd argument in range?
040665  303 01 0 00 000020                   CAILE   1,20
                                             JRST    [FMSGCD <? Range of 2901 registers is 0-20 (20=QReg)>
                                                      GO     FIOFF
040666  254 00 0 00 061151                            RTN]
040667  274 01 0 00 064626'                  SUB     1,RALUB                   ; see if range ok
                                             JUMPL   1,[FMSGCD <? Range nonsensical>
                                                         GO     FIOFF
040670  321 01 0 00 057436                               RTN]
040671  200 00 0 00 064626'                  MOVE    RALUB                     ; get 'from' argument
040672  202 00 0 00 040753                   MOVEM   ALUFR                     ; save it
040673  200 00 0 00 064627'                  MOVE    RALUE                     ; get 'to' argument
040674  202 00 0 00 040754                   MOVEM   ALUTO                     ; save it
040675  201 00 0 00 000010                   MOVEI   8                         ; get Z2901 flag
040676  202 00 0 00 064576'                  MOVEM   DFLAG                     ; save it
```

```

                                 ; Save CRAM location

 040677  200 01 0 00 061154              MOVE    1,[-1,,7750]              ; set up AOBJN word
 040700  260 17 0 00 044040              GO      SAVCRM                    ; save the CRAM location

                                 ; Now do each location

 040701  200 02 0 00 064626'     DZALU1: MOVE    2,RALUB                   ; get location to write
 040702  137 02 0 00 061203              DPB     2,[POINT 4,DZALUM+1,18]   ; set up register number
 040703  201 00 0 00 000002              MOVEI   2                         ; get destination field of 2 (BA)
 040704  306 02 0 00 000020              CAIN    2,20                      ; Q-Register?
 040705  201 00 0 00 000000              MOVEI   0                         ; yes - use destination of 0 (QF)
 040706  137 00 0 00 061204              DPB     [POINT 3,DZALUM+1,8]      ; set up destination field
 040707  201 01 0 00 040750              MOVEI   1,DZALUM                  ; set up microcode address
 040710  260 17 0 00 043207              GO      MLOADN                    ; load/verify it
 040711  255 00 0 00 000000              JFCL                              ; ignore error
 040712  350 00 0 00 000002              AOS     2                         ; point to next register
 040713  202 02 0 00 037755              MOVEM   2,ANEXT                   ; save next register
 040714  201 01 0 00 007750              MOVEI   1,7750                    ; get start address
 040715  242 01 0 00 000001              LSH     1,1                       ; position correctly
 040716  260 17 0 00 040621*             GO      LDRAR                     ; load the RAR
 040717  201 01 0 00 020010              MOVEI   1,MPRUN!SINCYC            ; start it up
 040720  260 17 0 00 040627*             GO      LDCSR
 040721  037 07 0 00 000003              TTALTM                            ; altmode key struck?
 040722  334 00 0 00 000000              SKIPA                             ; no - proceed
 040723  254 00 0 00 040727              JRST    DZALUX                    ; yes - exit
 040724  350 01 0 00 064626'             AOS     1,RALUB                   ; point to next location
 040725  317 01 0 00 064627'             CAMG    1,RALUE                   ; done yet?
 040726  254 00 0 00 040701              JRST    DZALU1                    ; no - loop till done

                                 ; Restore CRAM location and exit

 040727  200 01 0 00 061154      DZALUX: MOVE    1,[-1,,7750]              ; set up AOBJN word
 040730  260 17 0 00 044062              GO      RESCRM                    ; restore CRAM data
 040731  263 17 0 00 000000              RTN                               ; return
```

```

                                ; Microcode to examine 2901's

040732  775077  502004          DEALUM: MWORD   <ADDR=7750,JMAP,J=7750,SOA,OR,D=1,OENA,SELE,MGC=4>
040734  777777  777777                  -1

                                ; Microcode to deposit 2901's

040735  775077  510000          DDALUM: MWORD   <ADDR=7750,JMAP,J=7751,D=1>             ; 7750
040737  775177  530000                  MWORD   <CJP,J=7753,D=1,CENA,CCER>              ; 7751
040741  775277  510000                  MWORD   <JMAP,J=7751,D=1>                       ; 7752
040743  775300  000010                  MWORD   <CONT,SDO,OR,D=2,SELE,MGC=10>           ; 7753
040745  775477  540000                  MWORD   <JMAP,J=7754,D=1>                       ; 7754
040747  777777  777777                  -1

                                ; Microcode to zero 2901's

040750  775077  500000          DZALUM: MWORD   <ADDR=7750,JMAP,J=7750,SOA,AND,D=2>
040752  777777  777777                  -1

                                ; Arguments

040753  000000  000000          ALUFR:  0                               ; default range of
040754  000000  000020          ALUTO:  20                              ;   registers
```

```
                                    ;#*******************************************************************
                                    ;* DELOCS - Examine local storage
                                    ;
                                    ; Format:  ELOCS adr1,adr2
                                    ;#*******************************************************************

040755  200 00 0 00 041246          DELOCS: MOVE    LOCFR                   ; set up default of local storage range
040756  202 00 0 00 064632'                 MOVEM   RLOCB#                  ;   to be whatever was typed before
040757  200 00 0 00 041247                  MOVE    LOCTO                   ;   or if nothing typed, then 0-0
040760  202 00 0 00 064633'                 MOVEM   RLOCE#

                                    ; Get arguments

040761  200 01 0 00 061213                  MOVE    1,[FMSGCD <ELOCS addr1,addr2 (CR)>]
040762  260 17 0 00 040654*                 GO      .DARG                   ; get first address
040763  263 17 0 00 000000                   RTN                            ; error/altmode/question - exit
040764  254 00 0 00 040776                   JRST   DELOCO                  ; no arguments given - continue
040765  200 00 0 00 040657*                 MOVE    ARGUM                   ; get argument
040766  202 00 0 00 064632'                 MOVEM   RLOCB                   ; save in 'from' location
040767  202 00 0 00 064633'                 MOVEM   RLOCE                   ; save in 'to' location

040770  210 01 0 00 061213                  MOVN    1,[FMSGCD <ELOCS addr1,addr2 (CR)>]
040771  260 17 0 00 040762*                 GO      .DARG                   ; get 2nd address
040772  263 17 0 00 000000                   RTN                            ; error/altmode/question - exit
040773  254 00 0 00 040776                   JRST   DELOCO                  ; no arguments given - continue
040774  200 00 0 00 040765*                 MOVE    ARGUM                   ; get argument
040775  202 00 0 00 064633'                 MOVEM   RLOCE                   ; save in 'to' location

                                    ; Now range check the arguments

040776  331 01 0 00 064632'         DELOCO: SKIPL   1,RLOCB                 ; 1st argument in range?
040777  303 01 0 00 001777                  CAILE   1,1777
                                            JRST    [FMSGCD <? Range of local storage locations is 0-1777>
                                                     SETOM  ALTF
                                                     GO     FIOFF
041000  254 00 0 00 061226                           RTN]
041001  331 01 0 00 064633'                 SKIPL   1,RLOCE                 ; 2nd argument in range?
041002  303 01 0 00 001777                  CAILE   1,1777
                                            JRST    [FMSGCD <? Range of local storage locations is 0-1777>
                                                     GO     FIOFF
041003  254 00 0 00 061232                           RTN]
041004  274 01 0 00 064632'                 SUB     1,RLOCB                 ; see if range ok
                                            JUMPL   1,[FMSGCD <? Range nonsensical>
                                                       GO   FIOFF
041005  321 01 0 00 057436                             RTN]
041006  200 00 0 00 064632'                 MOVE    RLOCB                   ; get 'from' argument
041007  202 00 0 00 041246                  MOVEM   LOCFR                   ; save it
041010  200 00 0 00 064633'                 MOVE    RLOCE                   ; get 'to' argument
041011  202 00 0 00 041247                  MOVEM   LOCTO                   ; save it
041012  201 00 0 00 000011                  MOVEI   9                       ; get ELOCS flag
041013  202 00 0 00 064576'                 MOVEM   DFLAG                   ; save it
```

```

                                        ; Save CRAM locations

041014  200 01 0 00 061235                      MOVE    1,[-2,,7750]            ; set up AOBJN word
041015  260 17 0 00 044040                      GO      SAVCRM                  ; save the CRAM locations
041016  200 07 0 00 064632'                     MOVE    7,RLOCB                 ; get initial address

                                        ; Now do each location

041017  200 06 0 00 064632'             DELOC1: MOVE    6,RLOCB                 ; get current address
041020  274 06 0 00 000007                      SUB     6,7                     ; calculate word # (0..n)
041021  602 06 0 00 000003                      TRNE    6,3                     ; multiple of 4?
041022  037 00 0 00 053077                      TMSG    < >                     ; no - print a few spaces
041023  606 06 0 00 000003                      TRNN    6,3                     ; multiple of 4?
                                                GO      [TMSGC  <LS>            ; yes - start a new line
                                                         MOVE   RLOCB
                                                         PNTOCS
                                                         TMSG   </ >
041024  260 17 0 00 061237                               RTN]
041025  200 00 0 00 064632'                     MOVE    RLOCB                   ;  get local storage address
041026  137 00 0 00 061244                      DPB     [POINT 10,DELOCM,35]    ; save it in microword
041027  201 01 0 00 041223                      MOVEI   1,DELOCM                ; set up microcode address
041030  260 17 0 00 043207                      GO      MLOADN                  ; load/verify it
041031  255 00 0 00 000000                      JFCL                            ; ignore error
041032  201 01 0 00 007750                      MOVEI   1,7750                  ; get start address
041033  242 01 0 00 000001                      LSH     1,1                     ; position correctly
041034  260 17 0 00 040716*                     GO      LDRAR                   ; load the RAR
041035  201 01 0 00 000010                      MOVEI   1,MPRUN                 ; start it up
041036  260 17 0 00 040720*                     GO      LDCSR                   ; write to CSR register
041037  201 01 0 00 200000                      MOVEI   1,TSTEBF                ; set up to read EBUF data
041040  260 17 0 00 041036*                     GO      LDCSR                   ; write to CSR register
041041  260 17 0 00 040537*                     GO      RDEBUF                  ; read EBUF contents
041042  200 00 0 00 000001                      MOVE    1                       ; get register data
041043  037 13 0 00 000000                      PNTHW                           ; print it
041044  037 07 0 00 000003                      TTALTM                          ; altmode key struck?
041045  334 00 0 00 000000                      SKIPA                           ; no - proceed
                                                JRST    [SETOM  ALTF            ; yes - set altmode typed flag
                                                         SETZM  MULFLG          ;   and clear 'examine next' flag
041046  254 00 0 00 061245                               JRST   DELOCX]         ;   and exit
041047  350 01 0 00 064632'                     AOS     1,RLOCB                 ; point to next location
041050  202 01 0 00 037755                      MOVEM   1,ANEXT                 ; save next location
041051  317 01 0 00 064633'                     CAMG    1,RLOCE                 ; done yet?
041052  254 00 0 00 041017                      JRST    DELOC1                  ; no - loop till done

                                        ; Restore CRAM locations and exit

041053  336 00 0 00 064616'             DELOCX: SKIPN   MULFLG                  ; in 'examine next'?
041054  037 00 0 00 030242                      PCRL                            ; no - print blank line
041055  200 01 0 00 061235                      MOVE    1,[-2,,7750]            ; set up AOBJN word
041056  260 17 0 00 044062                      GO      RESCRM                  ; restore CRAM data
041057  263 17 0 00 000000                      RTN                             ; return
```

```
                                        ;#*****************************************************************
                                        ;* DDLOCS - Deposit local storage
                                        ;
                                        ; Format:  DLOCS adr,data
                                        ;#*****************************************************************

041060  200 01 0 00 061255              DDLOCS: MOVE    1,[FMSGCD <DLOCS addr,data (CR)>]
041061  260 17 0 00 040771*                     GO      .DARG                   ; get address
041062  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
                                                  JRST  [FMSGCD <? Missing argument>
                                                          GO      FIOFF
041063  254 00 0 00 055164                                RTN]
041064  200 00 0 00 040774*             DDLOCO: MOVE    ARGUM                   ; get argument
041065  301 00 0 00 000000                       CAIL    0                       ; in range 0-1777?
041066  303 00 0 00 001777                       CAILE   1777
                                                JRST    [FMSGCD <? Range of Local Storage addresses is 0-1777>
                                                          GO      FIOFF
041067  254 00 0 00 061270                                RTN]
041070  202 00 0 00 041246                      MOVEM   LOCFR                   ; save 'from' argument
041071  202 00 0 00 041247                      MOVEM   LOCTO                   ; save 'to' argument
041072  200 01 0 00 061255                      MOVE    1,[FMSGCD <DLOCS addr,data (CR)>]
041073  260 17 0 00 040573*                     GO      .DARG                   ; get data
041074  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
                                                  JRST  [FMSGCD <? Missing argument>
                                                          GO      FIOFF
041075  254 00 0 00 055164                                RTN]
041076  200 00 0 00 041064*                     MOVE    ARGUM                   ; get argument
041077  202 00 0 00 064607'                     MOVEM   LARG2#                  ; save it
041100  201 00 0 00 000012                      MOVEI   ^D10                    ; get DLOCS flag
041101  202 00 0 00 064576'                     MOVEM   DFLAG                   ; save it

                                        ; Save CRAM locations

041102  200 01 0 00 061172                      MOVE    1,[-5,,7750]            ; set up AOBJN word
041103  260 17 0 00 044040                      GO      SAVCRM                  ; save the CRAM locations

                                        ; Load microcode

041104  200 00 0 00 041246                      MOVE    LOCFR                   ; get location to write
041105  137 00 0 00 061273                      DPB     [POINT 10,DDLOCM+8,35]  ; set up local storage address
041106  202 00 0 00 037755                      MOVEM   ANEXT                   ; save address
041107  350 00 0 00 037755                      AOS     ANEXT                   ; set to next address
041110  201 01 0 00 041230                      MOVEI   1,DDLOCM                ; set up microcode address
041111  260 17 0 00 043207                      GO      MLOADN                  ; load/verify it
041112  255 00 0 00 000000                      JFCL                            ; ignore error
```

```


                                        ; Now do the deposit

 041113  201 01 0 00 007750                     MOVEI   1,7750                  ; get start address
 041114  242 01 0 00 000001                     LSH     1,1                     ; position correctly
 041115  260 17 0 00 041034*                    GO      LDRAR                   ; load the RAR
 041116  201 01 0 00 000010                     MOVEI   1,MPRUN                 ; set initial data
 041117  260 17 0 00 041040*                    GO      LDCSR                   ; start the port
 041120  200 01 0 00 064607'                    MOVE    1,LARG2                 ; get data to write
 041121  260 17 0 00 040625*                    GO      .DATAO                  ; give to port
 041122  400 01 0 00 000000                     SETZ    1,                      ; stop the port
 041123  260 17 0 00 041117*                    GO      LDCSR


                                        ; Check if additional arguments

 041124  336 00 0 00 040630*                    SKIPN   ARGFLG                  ; any more arguments
 041125  254 00 0 00 041131                     JRST    DDLOCX                  ; no - exit
 041126  200 00 0 00 037755                     MOVE    ANEXT                   ; get register number
 041127  202 00 0 00 041076*                    MOVEM   ARGUM                   ; save it
 041130  254 00 0 00 041064                     JRST    DDLOCO                  ; loop till done

                                        ; Restore CRAM locations and exit

 041131  200 01 0 00 061172             DDLOCX: MOVE    1,[-5,,7750]            ; set up AOBJN word
 041132  260 17 0 00 044062                     GO      RESCRM                  ; restore CRAM data
 041133  263 17 0 00 000000                     RTN                             ; return
```

```
                                      ;#*********************************************************************
                                      ;* DZLOCS - Zero local storage
                                      ;
                                      ; Format:  ZLOCS adr1,adr2
                                      ;#*********************************************************************

041134  200 00 0 00 041246            DZLOCS: MOVE    LOCFR                   ; set up default of local storage range
041135  202 00 0 00 064632'                   MOVEM   RLOCB#                  ;   to be whatever was typed before
041136  200 00 0 00 041247                    MOVE    LOCTO                   ;   or if nothing typed, then 0-0
041137  202 00 0 00 064633'                   MOVEM   RLOCE#

                                      ; Get arguments

041140  200 01 0 00 061302                    MOVE    1,[FMSGCD <ZLOCS addr1,addr2 (CR)>]
041141  260 17 0 00 041061*                   GO      .DARG                   ; get first address
041142  263 17 0 00 000000                     RTN                            ; error/altmode/question - exit
041143  254 00 0 00 041155                     JRST   DZLOCO                  ; no arguments given - continue
041144  200 00 0 00 041127*                   MOVE    ARGUM                   ; get argument
041145  202 00 0 00 064632'                   MOVEM   RLOCB                   ; save in 'from' location
041146  202 00 0 00 064633'                   MOVEM   RLOCE                   ; save in 'to' location

041147  210 01 0 00 061302                    MOVN    1,[FMSGCD <ZLOCS addr1,addr2 (CR)>]
041150  260 17 0 00 041141*                   GO      .DARG                   ; get 2nd address
041151  263 17 0 00 000000                     RTN                            ; error/altmode/question - exit
041152  254 00 0 00 041155                     JRST   DZLOCO                  ; no arguments given - continue
041153  200 00 0 00 041144*                   MOVE    ARGUM                   ; get argument
041154  202 00 0 00 064633'                   MOVEM   RLOCE                   ; save in 'to' location

                                      ; Now range check the arguments

041155  331 01 0 00 064632'           DZLOCO: SKIPL   1,RLOCB                 ; 1st argument in range?
041156  303 01 0 00 001777                    CAILE   1,1777
                                              JRST    [FMSGCD <? Range of local storage locations is 0-1777>
                                                        GO      FIOFF
041157  254 00 0 00 061232                              RTN]
041160  331 01 0 00 064633'                   SKIPL   1,RLOCE                 ; 2nd argument in range?
041161  303 01 0 00 001777                    CAILE   1,1777
                                              JRST    [FMSGCD <? Range of local storage locations is 0-1777>
                                                        GO      FIOFF
041162  254 00 0 00 061232                              RTN]
041163  274 01 0 00 064632'                   SUB     1,RLOCB                 ; see if range ok
                                              JUMPL   1,[FMSGCD <? Range nonsensical>
                                                          GO    FIOFF
041164  321 01 0 00 057436                                RTN]
041165  200 00 0 00 064632'                   MOVE    RLOCB                   ; get 'from' argument
041166  202 00 0 00 041246                    MOVEM   LOCFR                   ; save it
041167  200 00 0 00 064633'                   MOVE    RLOCE                   ; get 'to' argument
041170  202 00 0 00 041247                    MOVEM   LOCTO                   ; save it
041171  201 00 0 00 000013                    MOVEI   ^D11                    ; get ZLOCS flag
041172  202 00 0 00 064576'                   MOVEM   DFLAG                   ; save it
```

```


                                    ; Save CRAM location

041173  200 01 0 00 061154                  MOVE    1,[-1,,7750]                    ; set up AOBJN word
041174  260 17 0 00 044040                  GO      SAVCRM                          ; save the CRAM locations

                                    ; Now do each location

041175  200 00 0 00 064632'         DZLOC1: MOVE    RLOCB                           ; get location to write
041176  137 00 0 00 061303                  DPB     [POINT 10,DZLOCM,35]            ; set up local storage address
041177  201 01 0 00 041243                  MOVEI   1,DZLOCM                        ; set up microcode address
041200  260 17 0 00 043207                  GO      MLOADN                          ; load/verify it
041201  255 00 0 00 000000                  JFCL                                    ; ignore error
041202  201 01 0 00 007750                  MOVEI   1,7750                          ; get start address
041203  242 01 0 00 000001                  LSH     1,1                             ; position correctly
041204  260 17 0 00 041115*                 GO      LDRAR                           ; load the RAR
041205  201 01 0 00 000010                  MOVEI   1,MPRUN                         ; start it up
041206  260 17 0 00 041123*                 GO      LDCSR
041207  400 01 0 00 000000                  SETZ    1,                              ; stop the port
041210  260 17 0 00 041206*                 GO      LDCSR
041211  037 07 0 00 000003                  TTALTM                                  ; altmode key struck?
041212  334 00 0 00 000000                  SKIPA                                   ; no - proceed
041213  254 00 0 00 041220                  JRST    DZLOCX                          ; yes - exit
041214  350 01 0 00 064632'                 AOS     1,RLOCB                         ; point to next location
041215  202 01 0 00 037755                  MOVEM   1,ANEXT                         ; save next location
041216  317 01 0 00 064633'                 CAMG    1,RLOCE                         ; done yet?
041217  254 00 0 00 041175                  JRST    DZLOC1                          ; no - loop till done

                                    ; Restore CRAM location and exit

041220  200 01 0 00 061154          DZLOCX: MOVE    1,[-1,,7750]                    ; set up AOBJN word
041221  260 17 0 00 044062                  GO      RESCRM                          ; restore CRAM data
041222  263 17 0 00 000000                  RTN                                     ; return
```

```
 5542
 5543                                 ; Microcode to examine local storage
 5544
 5545  041223  775077  510000         DELOCM: MWORD   <ADDR=7750,JMAP,J=7751,SDO,OR,D=2,RDLM>
 5546  041225  775177  512004                 MWORD   <JMAP,J=7751,SOA,OR,D=1,OENA,SELE,MGC=4>
 5547  041227  777777  777777                 -1
 5548
 5549                                 ; Microcode to deposit local storage
 5550
 5551  041230  775077  510000         DDLOCM: MWORD   <ADDR=7750,JMAP,J=7751,D=1>             ; 7750
 5552  041232  775177  530000                 MWORD   <CJP,J=7753,D=1,CENA,CCER>              ; 7751
 5553  041234  775277  510000                 MWORD   <JMAP,J=7751,D=1>                       ; 7752
 5554  041236  775300  000010                 MWORD   <CONT,SDO,OR,D=2,SELE,MGC=10>           ; 7753
 5555  041240  775477  542000                 MWORD   <JMAP,J=7754,SOA,OR,D=1,OENA,LDLM,MGC=0>; 7754
 5556  041242  777777  777777                 -1
 5557
 5558                                 ; Microcode to zero local storage
 5559
 5560  041243  775077  502000         DZLOCM: MWORD   <ADDR=7750,JMAP,J=7750,SOA,AND,D=1,LDLM,OENA>
 5561  041245  777777  777777                 -1
 5562
 5563                                 ; Arguments
 5564
 5565  041246  000000  000000         LOCFR:  0                                       ; default range of
 5566  041247  000000  000000         LOCTO:  0                                       ;   locations
```

```
                                        ;#*********************************************************************
                                        ;* DMARK - Set mark bit in CRAM locations
                                        ;#*********************************************************************

                                        ; First check for any argument

 041250  336 00 0 00 041124*            DMARK:  SKIPN   ARGFLG                  ; argument given?
                                                JRST    [FMSGCD <? Missing argument>
                                                         GO      FIOFF
 041251  254 00 0 00 055164                              RTN]

                                        ; Obtain argument

 041252  200 01 0 00 061312             DMARK0: MOVE    1,[FMSGCD <MARK addr,addr,... (CR)>]
 041253  260 17 0 00 041073*                    GO      .OARG                   ; get argument
 041254  263 17 0 00 000000                      RTN                            ; error/altmode/question - exit
 041255  263 17 0 00 000000                      RTN                            ; done - no more arguments
 041256  331 01 0 00 041153*                    SKIPL   1,ARGUM                 ; argument out of range?
 041257  303 01 0 00 007777                     CAILE   1,7777                  ;   (0-7777)
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO      FIOFF
 041260  254 00 0 00 057070                              RTN]

                                        ; Then ensure the port is stopped

 041261  260 17 0 00 037742*                    GO      ISTOP                   ; stop the port

                                        ; Now obtain CRAM data and set MARK bit

 041262  202 01 0 00 037750                     MOVEM   1,CADDR                 ; set up address to load
 041263  476 00 0 00 037751                     SETOM   CWORDL                  ; initialize left half
 041264  476 00 0 00 037752                     SETOM   CWORDR                  ; initialize right half
 041265  260 17 0 00 040430                     GO      DRCRAM                  ; read the data
 041266  200 02 0 00 037751                     MOVE    2,CWORDL                ; get left half
 041267  200 03 0 00 037752                     MOVE    3,CWORDR                ; get right half
 041270  660 03 0 00 000001                     TRO     3,1                     ; set mark bit
 041271  202 03 0 00 037752                     MOVEM   3,CWORDR                ; save data
 041272  261 17 0 00 035764                     PUT     PARFLG                  ; save state of parity flag
 041273  402 00 0 00 035764                     SETZM   PARFLG                  ; clear parity flag
 041274  260 17 0 00 040376                     GO      DWCRAM                  ; load CRAM data
 041275  262 17 0 00 035764                     GET     PARFLG                  ; restore state of parity flag
 041276  254 00 0 00 041252                     JRST    DMARK0                  ; get next address
```

```
                                        ;#*********************************************************************
                                        ;* DRMARK - Remove mark bit in CRAM locations
                                        ;#*********************************************************************

                                        ; First check for any argument

041277  336 00 0 00 041250*             DRMARK: SKIPN   ARGFLG                          ; argument given?
                                                JRST    [FMSGCD <? Missing argument>
                                                         GO      FIOFF
041300  254 00 0 00 055164                               RTN]

                                        ; Obtain argument

041301  200 01 0 00 061321              DRMARO: MOVE    1,[FMSGCD <RMARK addr,addr,... (CR)>]
041302  260 17 0 00 041253*                     GO      .OARG                           ; get argument
041303  263 17 0 00 000000                       RTN                                    ; error/altmode/question - exit
041304  263 17 0 00 000000                       RTN                                    ; done - no more arguments
041305  331 01 0 00 041256*                     SKIPL   1,ARGUM                         ; argument out of range?
041306  303 01 0 00 007777                      CAILE   1,7777                          ;   (0-7777)
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO      FIOFF
041307  254 00 0 00 057070                               RTN]

                                        ; Then ensure the port is stopped

041310  260 17 0 00 041261*                     GO      ISTOP                           ; stop the port

                                        ; Now obtain CRAM data and set MARK bit

041311  202 01 0 00 037750                      MOVEM   1,CADDR                         ; set up address to load
041312  476 00 0 00 037751                      SETOM   CWORDL                          ; initialize left half
041313  476 00 0 00 037752                      SETOM   CWORDR                          ; initialize right half
041314  260 17 0 00 040430                      GO      DRCRAM                          ; read the data
041315  200 02 0 00 037751                      MOVE    2,CWORDL                        ; get left half
041316  200 03 0 00 037752                      MOVE    3,CWORDR                        ; get right half
041317  620 03 0 00 000001                      TRZ     3,1                             ; clear mark bit
041320  202 03 0 00 037752                      MOVEM   3,CWORDR                        ; save data
041321  261 17 0 00 035764                      PUT     PARFLG                          ; save state of parity flag
041322  402 00 0 00 035764                      SETZM   PARFLG                          ; clear parity flag
041323  260 17 0 00 040376                      GO      DWCRAM                          ; load CRAM data
041324  262 17 0 00 035764                      GET     PARFLG                          ; restore state of parity flag
041325  254 00 0 00 041301                      JRST    DRMARO                          ; get next address
```

```
                                        ;#**********************************************************************
                                        ;* DCMARK - Clear all mark bits
                                        ;
                                        ; If arguments are given, they are assumed to the the locations between
                                        ; which all mark bits are to be removed.
                                        ;#**********************************************************************

                                        ; Start to process the arguments typed - both should be CRAM
                                        ; addresses in octal.

041326  400 05 0 00 000000              DCMARK: SETZ    5,                      ; init range of addresses
041327  201 06 0 00 007777                      MOVEI   6,7777                  ;   to 0-7777

                                        ; Get first address

041330  200 01 0 00 061330                      MOVE    1,[FMSGCD <CMARK addr1,addr2 (CR)>]
041331  260 17 0 00 041302*                     GO      .OARG                   ; get first address
041332  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
041333  254 00 0 00 041345                        JRST  DCMAR0                  ; no arguments given - continue
041334  200 00 0 00 041305*                     MOVE    ARGUM                   ; get argument
041335  202 00 0 00 000005                      MOVEM   5                       ; save in 'from' location
041336  202 00 0 00 000006                      MOVEM   6                       ; save in 'to' location

                                        ; Get 2nd address

041337  210 01 0 00 061330                      MOVN    1,[FMSGCD <CMARK addr1,addr2 (CR)>]
041340  260 17 0 00 041331*                     GO      .OARG                   ; get first address
041341  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
041342  254 00 0 00 041345                        JRST  DCMAR0                  ; no arguments given - continue
041343  200 00 0 00 041334*                     MOVE    ARGUM                   ; get argument
041344  202 00 0 00 000006                      MOVEM   6                       ; save in 'to' location

                                        ; Now range check the arguments

041345  331 00 0 00 000005              DCMAR0: SKIPL   5                       ; 1st argument in range?
041346  303 05 0 00 007777                      CAILE   5,7777
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO      FIOFF
041347  254 00 0 00 057070                               RTN]
041350  331 01 0 00 000006                      SKIPL   1,6                     ; 2nd argument in range?
041351  303 06 0 00 007777                      CAILE   6,7777
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO      FIOFF
041352  254 00 0 00 057070                               RTN]
041353  274 01 0 00 000005                      SUB     1,5                     ; see if range ok
                                                JUMPL   1,[FMSGCD <? Range nonsensical>
                                                           GO    FIOFF
041354  321 01 0 00 057436                                 RTN]
```

```
                                         ; Then ensure the port is stopped

041355  260 17 0 00 041310*                    GO      ISTOP                   ; stop the port

                                         ; Now obtain CRAM data and load back into CRAM without MARK bit set

041356  202 05 0 00 037750               DCMAR1: MOVEM   5,CADDR                 ; set up address to load
041357  476 00 0 00 037751                       SETOM   CWORDL                  ; initialize left half
041360  476 00 0 00 037752                       SETOM   CWORDR                  ; initialize right half
041361  260 17 0 00 040430                       GO      DRCRAM                  ; read the data
041362  200 02 0 00 037751                       MOVE    2,CWORDL                ; get left half
041363  200 03 0 00 037752                       MOVE    3,CWORDR                ; get right half
041364  626 03 0 00 000001                       TRZN    3,1                     ; clear mark bit - set?
041365  254 00 0 00 041373                       JRST    DCMAR2                  ; no - continue
041366  202 03 0 00 037752                       MOVEM   3,CWORDR                ; save data
041367  261 17 0 00 035764                       PUT     PARFLG                  ; save state of parity flag
041370  402 00 0 00 035764                       SETZM   PARFLG                  ; clear parity flag
041371  260 17 0 00 040376                       GO      DWCRAM                  ; load CRAM data
041372  262 17 0 00 035764                       GET     PARFLG                  ; restore state of parity flag
041373  350 00 0 00 000005               DCMAR2: AOS     5                       ; increment 'from' address
041374  317 05 0 00 000006                       CAMG    5,6                     ; exceed 'to' address?
041375  254 00 0 00 041356                       JRST    DCMAR1                  ; no - get next address
041376  263 17 0 00 000000                       RTN                             ; yes - exit
```

```
                                   ;#***********************************************************************
                                   ;* DLMARK - List locations with mark bit set
                                   ;
                                   ; If arguments are given, they are assumed to the the locations between
                                   ; which all mark bits are to be listed.
                                   ;#***********************************************************************

                                   ; Start to process the arguments typed - both should be CRAM
                                   ; addresses in octal.

041377  400 05 0 00 000000         DLMARK: SETZ    5,                      ; init range of addresses
041400  201 06 0 00 007777                 MOVEI   6,7777                  ;   to 0-7777

                                   ; Get first address

041401  200 01 0 00 061337                 MOVE    1,[FMSGCD <LMARK addr1,addr2 (CR)>]
041402  260 17 0 00 041340*                GO      .OARG                   ; get first address
041403  263 17 0 00 000000                  RTN                            ; error/altmode/question - exit
041404  254 00 0 00 041416                  JRST   DLMARO                  ; no arguments given - continue
041405  200 00 0 00 041343*                MOVE    ARGUM                   ; get argument
041406  202 00 0 00 000005                 MOVEM   5                       ; save in 'from' location
041407  202 00 0 00 000006                 MOVEM   6                       ; save in 'to' location

                                   ; Get 2nd address

041410  210 01 0 00 061337                 MOVN    1,[FMSGCD <LMARK addr1,addr2 (CR)>]
041411  260 17 0 00 041402*                GO      .OARG                   ; get first address
041412  263 17 0 00 000000                  RTN                            ; error/altmode/question - exit
041413  254 00 0 00 041416                  JRST   DLMARO                  ; no arguments given - continue
041414  200 00 0 00 041405*                MOVE    ARGUM                   ; get argument
041415  202 00 0 00 000006                 MOVEM   6                       ; save in 'to' location

                                   ; Now range check the arguments

041416  331 00 0 00 000005         DLMARO: SKIPL   5                       ; 1st argument in range?
041417  303 05 0 00 007777                 CAILE   5,7777
                                           JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                    GO      FIOFF
041420  254 00 0 00 057070                          RTN]
041421  331 01 0 00 000006                 SKIPL   1,6                     ; 2nd argument in range?
041422  303 06 0 00 007777                 CAILE   6,7777
                                           JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                    GO      FIOFF
041423  254 00 0 00 057070                          RTN]
041424  274 01 0 00 000005                 SUB     1,5                     ; see if range ok
                                           JUMPL   1,[FMSGCD <? Range nonsensical>
                                                       GO   FIOFF
041425  321 01 0 00 057436                             RTN]
```

```

                                        ; Then ensure the port is stopped

041426  260 17 0 00 041355*                     GO      ISTOP                   ; stop the port
041427  200 07 0 00 061340                      MOVE    7,[-2]                  ; initialize mark bit count
041430  037 00 0 00 061341                      TMSGC   <Mark bit set at:  >

                                        ; Now obtain CRAM data and check mark bit

041431  202 05 0 00 037750              DLMAR1: MOVEM   5,CADDR                 ; set up address to load
041432  476 00 0 00 037751                      SETOM   CWORDL                  ; initialize left half
041433  476 00 0 00 037752                      SETOM   CWORDR                  ; initialize right half
041434  260 17 0 00 040430                      GO      DRCRAM                  ; read the data
041435  200 03 0 00 037752                      MOVE    3,CWORDR                ; get right word
041436  606 03 0 00 000001                      TRNN    3,1                     ; mark bit set?
041437  254 00 0 00 041445                      JRST    DLMAR2                  ; no - continue

                                        ; Print location

041440  350 00 0 00 000007                      AOS     7                       ; next 5 characters
041441  303 07 0 00 000007                      CAILE   7,7                     ; done with line?
                                                GO      [SETO   7,              ; yes - init counter
                                                         TMSGC  <
                                                                                  >
041442  260 17 0 00 061353                               RTN]
041443  001 04 0 00 000005                      PNTOCC  4,5                     ; print address
041444  037 00 0 00 000040                      PSP                             ; print a space

                                        ; Increment to next location

041445  037 07 0 00 000003              DLMAR2: TTALTM                          ; altmode key struck?
041446  334 00 0 00 000000                      SKIPA                           ; no - proceed
041447  254 00 0 00 041453                      JRST    DLMARX                  ; yes - exit
041450  350 00 0 00 000005                      AOS     5                       ; increment 'from' address
041451  317 05 0 00 000006                      CAMG    5,6                     ; exceed 'to' address?
041452  254 00 0 00 041431                      JRST    DLMAR1                  ; no - get next address
041453  037 00 0 00 030242              DLMARX: PCRL                            ; yes - print blank line
041454  263 17 0 00 000000                      RTN                             ;   and exit
```

```
                                 ;#*******************************************************************
                                 ;* DENEXT - Examine next
                                 ;#*******************************************************************

041455  402 00 0 00 041414*     DENEXT: SETZM   ARGUM                   ; default to repeat count
041456  350 00 0 00 041455*             AOS     ARGUM                   ;   of 1
041457  200 01 0 00 061366              MOVE    1,[FMSGCD (ENEXT repcount <CR> or ENEXT <CR>)]
041460  260 17 0 00 041150*             GO      .DARG                   ; get repeat count
041461  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
041462  255 00 0 00 000000               JFCL                           ; no argument given
041463  200 00 0 00 041456*             MOVE    ARGUM                   ; get argument

                                ; Do the examine

041464  476 00 0 00 064616'     DENEO:  SETOM   MULFLG                  ; set 'multiole examine' flag
041465  200 00 0 00 041463*             MOVE    ARGUM                   ; get argument
041466  307 00 0 00 000001              CAIG    1                       ; only one more repetition?
041467  402 00 0 00 064616'             SETZM   MULFLG                  ; yes - clear 'multiple examine' flag
041470  200 01 0 00 064576'             MOVE    1,DFLAG                 ; get last operation done
041471  554 00 0 01 041502              HLRZ    DETAB(1)                ; get new value of DFLAG
041472  202 00 0 00 064576'             MOVEM   DFLAG                   ; save it
041473  550 02 0 01 041502              HRRZ    2,DETAB(1)              ; get dispatch address
041474  260 17 0 02 000000              GO      (2)                     ; dispatch
041475  336 00 0 00 064570'             SKIPN   ALTF                    ; altmode typed?
041476  377 00 0 00 041465*             SOSG    ARGUM                   ; repeat count exnausted?
041477  263 17 0 00 000000              RTN                             ; yes - exit
041500  402 00 0 00 064617'             SETZM   MULINI                  ; clear 'initial examine' flag
041501  254 00 0 00 041464              JRST    DENEO                   ; no - loop till done

                                ; Dispatch table

041502  000000  041516          DETAB:  0,,DENEX0                       ; ECRAM
041503  000000  041516                  0,,DENEX0                       ; DCRAM
041504  000000  041516                  0,,DENEX0                       ; BCRAM
041505  000000  041516                  0,,DENEX0                       ; ACRAM
041506  000000  041516                  0,,DENEX0                       ; ZCRAM
041507  000005  041516                  5,,DENEX0                       ; LCRAM
041510  000006  041522                  6,,DENEX1                       ; E2901
041511  000006  041522                  6,,DENEX1                       ; D2901
041512  000006  041522                  6,,DENEX1                       ; Z2901
041513  000011  041526                  9,,DENEX2                       ; ELOCS
041514  000011  041526                  9,,DENEX2                       ; DLOCS
041515  000011  041526                  9,,DENEX2                       ; ZLOCS
```

```



                                   ; To handle CRAM examine ...

041516  200 00 0 00 037755         DENEXO: MOVE    ANEXT                   ; get address
041517  202 00 0 00 037747                 MOVEM   CRAMTO                  ; set up 'to' and 'from'
041520  202 00 0 00 037746                 MOVEM   CRAMFR                  ;   addresses
041521  254 00 0 00 037726                 PJRST   DCOMO                   ; go handle

                                   ; To handle 2901 registers

041522  200 00 0 00 037755         DENEX1: MOVE    ANEXT                   ; get register number
041523  202 00 0 00 064626'                MOVEM   RALUB                   ; save in 'from' location
041524  202 00 0 00 064627'                MOVEM   RALUE                   ; save in 'to' location
041525  254 00 0 00 040472                 PJRST   DEALUO                  ; go handle

                                   ; To handle local storage

041526  200 00 0 00 037755         DENEX2: MOVE    ANEXT                   ; get address
041527  202 00 0 00 064632'                MOVEM   RLOCB                   ; save in 'from' location
041530  202 00 0 00 064633'                MOVEM   RLOCE                   ; save in 'to' location
041531  254 00 0 00 040776                 PJRST   DELOCO                  ; go handle
```

```
                                        ;#*********************************************************************
                                        ;* DDNEXT - Deposit next
                                        ;#*********************************************************************

041532  200 01 0 00 064576'             DDNEXT: MOVE    1,DFLAG                 ; get last operation done
041533  554 00 0 01 041537                      HLRZ    DDTAB(1)                ; get new value of DFLAG
041534  202 00 0 00 064576'                     MOVEM   DFLAG                   ; save it
041535  550 02 0 01 041537                      HRRZ    2,DDTAB(1)              ; get dispatch address
041536  254 00 0 02 000000                      JRST    (2)                     ; dispatch

                                        ; Dispatch table

041537  000001  041553                  DDTAB:  1,,DDNEX0                       ; ECRAM
041540  000001  041553                          1,,DDNEX0                       ; DCRAM
041541  000002  041553                          2,,DDNEX0                       ; BCRAM
041542  000003  041553                          3,,DDNEX0                       ; ACRAM
041543  000003  041553                          3,,DDNEX0                       ; ZCRAM
041544  000003  041553                          3,,DDNEX0                       ; LCRAM
041545  000007  041562                          7,,DDNEX1                       ; E2901
041546  000007  041562                          7,,DDNEX1                       ; D2901
041547  000007  041562                          7,,DDNEX1                       ; Z2901
041550  000012  041567                          ^D10,,DDNEX2                    ; ELOCS
041551  000012  041567                          ^D10,,DDNEX2                    ; DLOCS
041552  000012  041567                          ^D10,,DDNEX2                    ; ZLOCS

                                        ; To handle CRAM deposit ...

041553  200 01 0 00 061372              DDNEX0: MOVE    1,[FMSGCD (DNEXT <CR>)]
041554  260 17 0 00 037654*                     GO      CHKARG                  ; check for argument
041555  263 17 0 00 000000                      RTN                             ; error - exit
041556  200 00 0 00 037755                      MOVE    ANEXT                   ; get address
041557  202 00 0 00 037747                      MOVEM   CRAMTO                  ; set up 'to' and 'from'
041560  202 00 0 00 037746                      MOVEM   CRAMFR                  ;   addresses
041561  254 00 0 00 037726                      PJRST   DCO                     ; go handle

                                        ; To handle 2901 registers

041562  200 00 0 00 037755              DDNEX1: MOVE    ANEX                    ; get register number
041563  202 00 0 00 064626'                     MOVEM   RALU                    ; save in 'from' location
041564  202 00 0 00 064627'                     MOVEM   RALU                    ; save in 'to' location
041565  202 00 0 00 041476*                     MOVEM   ARGU                    ; save it
041566  254 00 0 00 040563                      PJRST   DDAL )                  ; go handle

                                        ; To handle local storage

041567  200 00 0 00 037755              DDNEX2: MOVE    ANEXT                   ; get address
041570  202 00 0 00 064632'                     MOVEM   RLOCB                   ; save in 'from' location
041571  202 00 0 00 064633'                     MOVEM   RLOCE                   ; save in 'to' location
041572  202 00 0 00 041565*                     MOVEM   ARGUM                   ; save it
041573  254 00 0 00 041064                      PJRST   DDLOC0                  ; go handle
```

```

                                     ;*****************************************************************
                                     ;* DESTAK - examine 2910 stack
                                     ;*****************************************************************

041574  200 01 0 00 061377           DESTAK: MOVE    1,[FMSGCD (ESTACK <CR>)]
041575  260 17 0 00 041554*                  GO      CHKARG                  ; check for argument
041576  263 17 0 00 000000                   RTN                             ; error - exit
041577  200 01 0 00 061154                   MOVE    1,[-1,,7750]            ; set up AOBJN word
041600  260 17 0 00 044040                   GO      SAVCRM                  ; save the CRAM locations
041601  201 01 0 00 041631                   MOVEI   1,DESTKM                ; set up microcode address
041602  260 17 0 00 043207                   GO      MLOADN                  ; load/verify it
041603  255 00 0 00 000000                   JFCL                            ; ignore error

                                     ; Do the task

041604  260 17 0 00 041426*                  GO      ISTOP                   ; stop the port
041605  037 00 0 00 061400                   TMSGC   <2910 Stack:>
041606  037 00 0 00 061403                   TMSGC   < Location   Contents (addr)>
041607  474 04 0 00 000000                   SETO    4,                      ; initialize location pointer
041610  350 00 0 00 000004           DESTA0: AOS     4                       ; point to next location
041611  303 04 0 00 000004                   CAILE   4,4                     ; done yet?
041612  254 00 0 00 041625                   JRST    DESTAX                  ; yes - exit
041613  402 00 0 00 037562*                  SETZM   SDATA                   ; get initial CSR data
041614  201 00 0 00 007750                   MOVEI   7750                    ; set up next address
041615  202 00 0 00 037537*                  MOVEM   SNEXT                   ;   for IPASST
041616  260 17 0 00 036223*                  GO      IPASST                  ; single step once
                                               JRST    [FMSGCD <? Error accessing CSR register>
                                                        GO      FIOFF
041617  254 00 0 00 061411                              JRST    DESTAY]
                                               JRST    [FMSGCD <? Port not stopped>
                                                        GO      FIOFF
041620  254 00 0 00 061414                              JRST    DESTAY]
041621  255 00 0 00 000000                     JFCL                            ; error bits set in CSR
                                             XCT     [TMSGC  <   Top  >
                                                        TMSGC  <    -1  >
                                                        TMSGC  <    -2  >
                                                        TMSGC  <    -3  >
041622  256 00 0 04 061436                              TMSGC  <    -4  >](4)
041623  001 12 0 00 041615*                  PNTOCC  12,SNEXT                ; print address
041624  254 00 0 00 041610                   JRST    DESTA0                  ; loop till done

                                     ; Restore CRAM location/exit

041625  037 00 0 00 030242           DESTAX: PCRL
041626  200 01 0 00 061154           DESTAY: MOVE    1,[-1,,7750]            ; set up AOBJN word
041627  260 17 0 00 044062                   GO      RESCRM                  ; restore CRAM data
041630  263 17 0 00 000000                   RTN                             ; return

                                     ; Microcode

041631  775000  000000               DESTKM: MWORD   <ADDR=7750,CRTN,D=1>    ; 7750
041633  777777  777777                       -1
```

```
                                        ;#*****************************************************************
                                        ;* DBREAK - Set breakpoint
                                        ;
                                        ; A breakpoint is set by reading the particular CRAM location  and
                                        ; rewriting it with bad parity.  Nothing can be done at the break-
                                        ; point except read the CSR register.
                                        ;#*****************************************************************

                                        ; First check for any argument

041634  336 00 0 00 041277*             DBREAK: SKIPN   ARGFLG                  ; argument given?
                                                JRST    [FMSGCD <? Missing argument>
                                                         GO      FIOFF
041635  254 00 0 00 055164                               RTN]

                                        ; Obtain argument

041636  200 01 0 00 061451              DBREA0: MOVE    1,[FMSGCD <BREAK addr,addr,... (CR)>]
041637  260 17 0 00 041411*                     GO      .OARG                   ; get argument
041640  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
041641  263 17 0 00 000000                       RTN                            ; done - no more arguments
041642  331 01 0 00 041572*                     SKIPL   1,ARGUM                 ; argument out of range?
041643  303 01 0 00 007777                      CAILE   1,7777                  ;   (0-7777)
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO      FIOFF
041644  254 00 0 00 057070                               RTN]

                                        ; Then ensure the port is stopped

041645  260 17 0 00 041604*                     GO      ISTOP                   ; stop the port

                                        ; Now obtain CRAM data and give it bad parity

041646  202 01 0 00 037750                      MOVEM   1,CADDR                 ; set up address to load
041647  476 00 0 00 037751                      SETOM   CWORDL                  ; initialize left half
041650  476 00 0 00 037752                      SETOM   CWORDR                  ; initialize right half
041651  260 17 0 00 040430                      GO      DRCRAM                  ; read the data
041652  261 17 0 00 000001                      PUT     1                       ; save AC1
041653  474 01 0 00 000000                      SETO    1,                      ; set 'force bad parity' flag
041654  200 02 0 00 037751                      MOVE    2,CWORDL                ; get left half
041655  200 03 0 00 037752                      MOVE    3,CWORDR                ; get right half
041656  260 17 0 00 043625                      GO      CALPAR                  ; insert parity
041657  202 02 0 00 037751                      MOVEM   2,CWORDL                ; save left half
041660  202 03 0 00 037752                      MOVEM   3,CWORDR                ; save right half
041661  262 17 0 00 000001                      GET     1                       ; restore AC1
041662  261 17 0 00 035764                      PUT     PARFLG                  ; save state of parity flag
041663  402 00 0 00 035764                      SETZM   PARFLG                  ; clear parity flag
041664  260 17 0 00 040376                      GO      DWCRAM                  ; load CRAM data (now with bad parity)
041665  262 17 0 00 035764                      GET     PARFLG                  ; restore state of parity flag
041666  254 00 0 00 041636                      JRST    DBREA0                  ; get next address
```

```
                                        ;#*******************************************************************
                                        ;* DRBRK - Remove breakpoint
                                        ;#*******************************************************************

                                        ; First check for any argument

        041667  336 00 0 00 041634*     DRBRK:  SKIPN   ARGFLG                  ; argument given?
                                                JRST    [FMSGCD <? Missing argument>
                                                         GO     FIOFF
        041670  254 00 0 00 055164                       RTN]

                                        ; Obtain argument

        041671  200 01 0 00 061460      DRBRO:  MOVE    1,[FMSGCD <RBREAK addr,addr,... (CR)>]
        041672  260 17 0 00 041637*             GO      .OARG                   ; get argument
        041673  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
        041674  263 17 0 00 000000               RTN                            ; done - no more arguments
        041675  331 01 0 00 041642*             SKIPL   1,ARGUM                 ; argument out of range?
        041676  303 01 0 00 007777              CAILE   1,7777                  ;   (0-7777)
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO     FIOFF
        041677  254 00 0 00 057070                       RTN]

                                        ; Then ensure the port is stopped

        041700  260 17 0 00 041645*             GO      ISTOP                   ; stop the port

                                        ; Now obtain CRAM data and load back into CRAM with good parity

        041701  202 01 0 00 037750              MOVEM   1,CADDR                 ; set up address to load
        041702  476 00 0 00 037751              SETOM   CWORDL                  ; initialize left half
        041703  476 00 0 00 037752              SETOM   CWORDR                  ; initialize right half
        041704  260 17 0 00 040430              GO      DRCRAM                  ; read the data
        041705  261 17 0 00 035764              PUT     PARFLG                  ; save state of parity flag
        041706  476 00 0 00 035764              SETOM   PARFLG                  ; set parity flag
        041707  260 17 0 00 040376              GO      DWCRAM                  ; load CRAM data (now with good parity)
        041710  262 17 0 00 035764              GET     PARFLG                  ; restore state of parity flag
        041711  254 00 0 00 041671              JRST    DRBRO                   ; get next address
```

```
                                   ;#*******************************************************************
                                   ;* DCBRK - Clear all breakpoints
                                   ;
                                   ; If arguments are given, they are assumed to the the locations between
                                   ; which all breakpoints are to be removed.
                                   ;#*******************************************************************

                                   ; Start to process the arguments typed - both should be CRAM
                                   ; addresses in octal.

041712  400 05 0 00 000000         DCBRK:  SETZ    5,                      ; init range of addresses
041713  201 06 0 00 007777                 MOVEI   6,7777                  ;   to 0-7777

                                   ; Get first address

041714  200 01 0 00 061467                 MOVE    1,[FMSGCD <CBREAK addr1,addr2 (CR)>]
041715  260 17 0 00 041672*                GO      .OARG                   ; get first address
041716  263 17 0 00 000000                  RTN                            ; error/altmode/question - exit
041717  254 00 0 00 041731                  JRST   DCBRO                   ; no arguments given - continue
041720  200 00 0 00 041675*                MOVE    ARGUM                   ; get argument
041721  202 00 0 00 000005                 MOVEM   5                       ; save in 'from' location
041722  202 00 0 00 000006                 MOVEM   6                       ; save in 'to' location

                                   ; Get 2nd address

041723  210 01 0 00 061467                 MOVN    1,[FMSGCD <CBREAK addr1,addr2 (CR)>]
041724  260 17 0 00 041715*                GO      .OARG                   ; get first address
041725  263 17 0 00 000000                  RTN                            ; error/altmode/question - exit
041726  254 00 0 00 041731                  JRST   DCBRO                   ; no arguments given - continue
041727  200 00 0 00 041720*                MOVE    ARGUM                   ; get argument
041730  202 00 0 00 000006                 MOVEM   6                       ; save in 'to' location

                                   ; Now range check the arguments

041731  331 00 0 00 000005         DCBRO:  SKIPL   5                       ; 1st argument in range?
041732  303 05 0 00 007777                 CAILE   5,7777
                                           JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                     GO     FIOFF
041733  254 00 0 00 057070                           RTN]
041734  331 01 0 00 000006                 SKIPL   1,6                     ; 2nd argument in range?
041735  303 06 0 00 007777                 CAILE   6,7777
                                           JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                     GO     FIOFF
041736  254 00 0 00 057070                           RTN]
041737  274 01 0 00 000005                 SUB     1,5                     ; see if range ok
                                           JUMPL   1,[FMSGCD <? Range nonsensical>
                                                       GO    FIOFF
041740  321 01 0 00 057436                             RTN]
```

```

                                        ; Then ensure the port is stopped

        041741  260 17 0 00 041700*             GO      ISTOP                   ; stop the port

                                        ; Now obtain CRAM data and load back into CRAM with good parity

        041742  202 05 0 00 037750      DCBR1:  MOVEM   5,CADDR                 ; set up address to load
        041743  476 00 0 00 037751              SETOM   CWORDL                  ; initialize left half
        041744  476 00 0 00 037752              SETOM   CWORDR                  ; initialize right half
        041745  260 17 0 00 040430              GO      DRCRAM                  ; read the data
        041746  261 17 0 00 035764              PUT     PARFLG                  ; save state of parity flag
        041747  476 00 0 00 035764              SETOM   PARFLG                  ; set parity flag
        041750  260 17 0 00 040376              GO      DWCRAM                  ; load CRAM data (now with good parity)
        041751  262 17 0 00 035764              GET     PARFLG                  ; restore state of parity flag
        041752  350 00 0 00 000005              AOS     5                       ; increment 'from' address
        041753  317 05 0 00 000006              CAMG    5,6                     ; exceed 'to' address?
        041754  254 00 0 00 041742              JRST    DCBR1                   ; no - get next address
        041755  263 17 0 00 000000              RTN                             ; yes - exit
```

```

                                ;#*********************************************************************
                                ;* DLBRK - List breakpoints
                                ;
                                ; If arguments are given, they are assumed to the the locations between
                                ; which all breakpoints are to be listed.
                                ;#*********************************************************************

                                ; Start to process the arguments typed - both should be CRAM
                                ; addresses in octal.

041756  400 05 0 00 000000      DLBRK:  SETZ    5,                      ; init range of addresses
041757  201 06 0 00 007777              MOVEI   6,7777                  ;   to 0-7777

                                ; Get first address

041760  200 01 0 00 061476              MOVE    1,[FMSGCD <LBREAK addr1,addr2 (CR)>]
041761  260 17 0 00 041724*             GO      .OARG                   ; get first address
041762  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
041763  254 00 0 00 041775               JRST   DLBRO                   ; no arguments given - continue
041764  200 00 0 00 041727*             MOVE    ARGUM                   ; get argument
041765  202 00 0 00 000005              MOVEM   5                       ; save in 'from' location
041766  202 00 0 00 000006              MOVEM   6                       ; save in 'to' location

                                ; Get 2nd address

041767  210 01 0 00 061476              MOVN    1,[FMSGCD <LBREAK addr1,addr2 (CR)>]
041770  260 17 0 00 041761*             GO      .OARG                   ; get first address
041771  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
041772  254 00 0 00 041775               JRST   DLBRO                   ; no arguments given - continue
041773  200 00 0 00 041764*             MOVE    ARGUM                   ; get argument
041774  202 00 0 00 000006              MOVEM   6                       ; save in 'to' location

                                ; Now range check the arguments

041775  331 00 0 00 000005      DLBRO:  SKIPL   5                       ; 1st argument in range?
041776  303 05 0 00 007777              CAILE   5,7777
                                        JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                 GO     FIOFF
041777  254 00 0 00 057070                       RTN]
042000  331 01 0 00 000006              SKIPL   1,6                     ; 2nd argument in range?
042001  303 06 0 00 007777              CAILE   6,7777
                                        JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                 GO     FIOFF
042002  254 00 0 00 057070                       RTN]
042003  274 01 0 00 000005              SUB     1,5                     ; see if range ok
                                        JUMPL   1,[FMSGCD <? Range nonsensical>
                                                     GO    FIOFF
042004  321 01 0 00 057436                           RTN]
```

```

                                        ; Then ensure the port is stopped

 042005  260 17 0 00 041741*                     GO      ISTOP                           ; stop the port
 042006  200 07 0 00 061340                      MOVE    7,[-2]                          ; initialize breakpoint count
 042007  037 00 0 00 061477                      TMSGC   <Breakpoints:  >

                                        ; Now obtain CRAM data and check parity

 042010  202 05 0 00 037750             DLBR1:   MOVEM   5,CADDR                         ; set up address to load
 042011  476 00 0 00 037751                      SETOM   CWORDL                          ; initialize left half
 042012  476 00 0 00 037752                      SETOM   CWORDR                          ; initialize right half
 042013  260 17 0 00 040430                      GO      DRCRAM                          ; read the data
 042014  400 01 0 00 000000                      SETZ    1,                              ; clear 'bad' parity flag
 042015  200 02 0 00 037751                      MOVE    2,CWORDL                        ; get left word
 042016  200 03 0 00 037752                      MOVE    3,CWORDR                        ; get right word
 042017  260 17 0 00 043625                      GO      CALPAR                          ; calculate parity
 042020  316 02 0 00 037751                      CAMN    2,CWORDL                        ; same as before?
 042021  254 00 0 00 042027                      JRST    DLBR2                           ; yes - good parity

                                        ; Print location

 042022  350 00 0 00 000007                      AOS     7                               ; next 5 characters
 042023  303 07 0 00 000007                      CAILE   7,7                             ; done with line?
                                                 GO      [SETO   7,                      ; yes - init counter
                                                           TMSGC  <                     >
                                                           RTN]
 042024  260 17 0 00 061507
 042025  001 04 0 00 000005                      PNTOCC  4,5                             ; print address
 042026  037 00 0 00 000040                      PSP                                     ; print a space

                                        ; Increment to next location

 042027  037 07 0 00 000003             DLBR2:   TTALTM                                  ; altmode key struck?
 042030  334 00 0 00 000000                      SKIPA                                   ; no - proceed
 042031  254 00 0 00 042035                      JRST    DLBRX                           ; yes - exit
 042032  350 00 0 00 000005                      AOS     5                               ; increment 'from' address
 042033  317 05 0 00 000006                      CAMG    5,6                             ; exceed 'to' address?
 042034  254 00 0 00 042010                      JRST    DLBR1                           ; no - get next address
 042035  037 00 0 00 030242             DLBRX:   PCRL                                    ; yes - print blank line
 042036  263 17 0 00 000000                      RTN                                     ;   and exit
```

```

                                   ;#*********************************************************************
                                   ;* DLOAD - Load microcode into CRAM from a test.
                                   ;#*********************************************************************

        042037  210 01 0 00 061516     DLOAD:  MOVN    1,[FMSGCD (LOAD XXnnn <CR>)]
        042040  260 17 0 00 035631*            GO      .SARG                           ; get argument
        042041  263 17 0 00 000000              RTN                                    ; error/altmode/question - exit
                                                JRST   [FMSGCD <? Missing argument>
                                                        GO      FIOFF
        042042  254 00 0 00 055164                      RTN]
        042043  200 01 0 00 041773*            MOVE    1,ARGUM                         ; get argument
        042044  260 17 0 00 034315             GO      TSTDEC                          ; check for alternate format
                                                JRST   [FMSGCD <? Argument error>
                                                        GO      FIOFF
        042045  254 00 0 00 061517                      RTN]
                                                JRST   [FMSGC  <? Test number out of range 1 - >
                                                        MOVE    1
                                                        PNTOCF
                                                        PCRLF
                                                        GO      FIOFF
        042046  254 00 0 00 061522                      RTN]
        042047  260 17 0 00 042775             GO      TSTGET                          ; get test data
        042050  200 01 0 00 047016             MOVE    1,TSTMIC                        ; get test microcode address
                                               JUMPLE  1,[FMSGCD <? Test specified does not have microcode>
                                                          GO    FIOFF
        042051  323 01 0 00 061541                        RTN]
        042052  260 17 0 00 042005*            GO      ISTOP                           ; stop the port
        042053  260 17 0 00 043211             GO      MLOADY                          ; load it
        042054  255 00 0 00 000000              JFCL                                   ; failed
        042055  037 00 0 00 061544             TMSGC   <[Number of CRAM locations loaded/verified = >
        042056  200 00 0 00 064615'            MOVE    MLNUM                           ; get number of words loaded/verified
        042057  037 15 0 00 000000             PNTDEC
        042060  037 00 0 00 061556             TMSGD   <.]>
        042061  263 17 0 00 000000             RTN
```

```

                                                 ;#*********************************************************************
                                                 ;* DLIST - List microcode from test
                                                 ;#*********************************************************************

042062  210 01 0 00 061563    DLIST:  MOVN    1,[FMSGCD (LIST XXnnn <CR>)]
042063  260 17 0 00 042040*           GO      .SARG                   ; get argument
042064  263 17 0 00 000000             RTN                            ; error/altmode/question - exit
                                       JRST   [FMSGCD <? Missing argument>
                                               GO      FIOFF
042065  254 00 0 00 055164                     RTN]
042066  200 01 0 00 042043*           MOVE    1,ARGUM                 ; get argument
042067  260 17 0 00 034315            GO      TSTDEC                  ; check for alternate format
                                       JRST   [FMSGCD <? Argument error>
                                               GO      FIOFF
042070  254 00 0 00 061517                     RTN]
                                       JRST   [FMSGC  <? Test number out of range 1 - >
                                               MOVE    1
                                               PNTOCF
                                               PCRLF
                                               GO      FIOFF
042071  254 00 0 00 061522                     RTN]
042072  260 17 0 00 042775            GO      TSTGET                  ; get test data
042073  200 01 0 00 047016            MOVE    1,TSTMIC                ; get test microcode address
                                      JUMPLE  1,[FMSGCD <? Test specified does not have microcode>
                                                 GO    FIOFF
042074  323 01 0 00 061541                       RTN]
042075  260 17 0 00 043446            GO      MLIST                   ; list it
042076  263 17 0 00 000000            RTN                             ; continue
```

```
                                        ;#*********************************************************************
                                        ;* DFLOAD - Load microcode from file into CRAM
                                        ;#*********************************************************************

042077  200 01 0 00 061567              DFLOAD: MOVE    1,[FMSGCD (FLOAD <CR>)]
042100  260 17 0 00 041575*                     GO      CHKARG                  ; check for argument
042101  263 17 0 00 000000                      RTN                             ; error - exit
042102  400 02 0 00 000000                      SETZ    2,                      ; init count of words loaded
042103  260 17 0 00 000000*                     GO      FINIT                   ; init file for reading
042104  263 17 0 00 000000                       RTN                            ; failed - return
042105  260 17 0 00 042052*                     GO      ISTOP                   ; stop the port
042106  260 17 0 00 000000*             DFLOA0: GO      FGETW                   ; get microcode entry
042107  254 00 0 00 042121                       JRST   DFLOAX                  ; error/EOF occurred
042110  261 17 0 00 035764                      PUT     PARFLG                  ; save state of parity flag
042111  402 00 0 00 035764                      SETZM   PARFLG                  ; ensure no parity is generated
042112  260 17 0 00 040376                      GO      DWCRAM                  ; load it
042113  262 17 0 00 035764                      GET     PARFLG                  ; restore state of parity flag
042114  350 00 0 00 000002                      AOS     2                       ; increment count
042115  332 00 0 00 064570'                     SKIPE   ALTF                    ; altmode typed?
042116  254 00 0 00 042121                      JRST    DFLOAX                  ; yes - exit
042117  037 07 0 00 000003                      TTALTM                          ; altmode key struck?
042120  254 00 0 00 042106                      JRST    DFLOA0                  ; no - loop til done

042121  037 00 0 00 061570              DFLOAX: TMSGC   <[Number of CRAM locations loaded = >
042122  200 00 0 00 000002                      MOVE    2
042123  037 15 0 00 000000                      PNTDEC
042124  037 00 0 00 061556                      TMSGD   <.]>
042125  037 00 0 00 061600                      TMSGC   <[Ucode version >
042126  201 00 0 00 000137                      MOVEI   137                     ; read location 137 which contains
042127  202 00 0 00 037750                      MOVEM   CADDR                   ;   microcode version in MGC field
042130  260 17 0 00 040430                      GO      DRCRAM                  ;   bits 14-23
042131  135 00 0 00 060377                      LDB     [POINT 10,CWORDL,29]
042132  037 16 0 00 000003                      PNTOCS                          ; print it
042133  037 00 0 00 054523                      TMSGD   <]>
042134  263 17 0 00 000000                      RTN                             ; return
```

```
                                        ;#*******************************************************************
                                        ;* DFVERF - Verify microcode from file versus CRAM
                                        ;#*******************************************************************

 042135   200 01 0 00 061610            DFVERF: MOVE    1,[FMSGCD (FVERFY <CR>)]
 042136   260 17 0 00 042100*                   GO      CHKARG                  ; check for argument
 042137   263 17 0 00 000000                    RTN                             ; error - exit
 042140   400 02 0 00 000000                    SETZ    2,                      ; init count of words verified
 042141   402 00 0 00 064621'                   SETZM   MVNUM                   ; init error count
 042142   260 17 0 00 042103*                   GO      FINIT                   ; init file for reading
 042143   263 17 0 00 000000                     RTN                            ; failed - return
 042144   260 17 0 00 042105*                   GO      ISTOP                   ; stop the port
 042145   260 17 0 00 042106*           DFVERO: GO      FGETW                   ; get microcode entry
 042146   254 00 0 00 042167                     JRST   DFVERX                  ; error/EOF occurred
 042147   200 00 0 00 037751                    MOVE    CWORDL                  ; get left word
 042150   202 00 0 00 064655'                   MOVEM   XWORDL#                 ; save it
 042151   200 00 0 00 037752                    MOVE    CWORDR                  ; get right word
 042152   202 00 0 00 064656'                   MOVEM   XWORDR#                 ; save it
 042153   260 17 0 00 040430                    GO      DRCRAM                  ; read the location
 042154   200 00 0 00 037751                    MOVE    CWORDL                  ; get left word
 042155   312 00 0 00 064655'                   CAME    XWORDL#                 ; same as loaded?
 042156   254 00 0 00 042161                    JRST    .+3                     ; no - report error
 042157   200 00 0 00 037752                    MOVE    CWORDR                  ; get right word
 042160   312 00 0 00 064656'                   CAME    XWORDR#                 ; same as loaded?
 042161   260 17 0 00 042177                    GO      DFVERR                  ; no - report error
 042162   350 00 0 00 000002                    AOS     2                       ; increment count
 042163   332 00 0 00 064570'                   SKIPE   ALTF                    ; altmode typed?
 042164   254 00 0 00 042167                    JRST    DFVERX                  ; yes - exit
 042165   037 07 0 00 000003                    TTALTM                          ; altmode key struck?
 042166   254 00 0 00 042145                    JRST    DFVERO                  ; no - loop til done

 042167   037 00 0 00 061611            DFVERX: TMSGC   <[Number of CRAM locations verified = >
 042170   200 00 0 00 000002                    MOVE    2
 042171   037 15 0 00 000000                    PNTDEC
 042172   037 00 0 00 061621                    TMSG    <.  Errors = >
 042173   200 00 0 00 064621'                   MOVE    MVNUM
 042174   037 15 0 00 000000                    PNTDEC
 042175   037 00 0 00 061556                    TMSGD   <.]>
 042176   263 17 0 00 000000                    RTN                             ; return

                                        ; Print error data

 042177   261 17 0 00 000000            DFVERR: RPUT    (0,1,2,3)               ; save some AC's
 042203   350 01 0 00 064621'                   AOS     1,MVNUM                 ; get error count
 042204   303 01 0 00 000004                    CAILE   1,4                     ; more than 4 errors?
 042205   254 00 0 00 042222                    JRST    DFVERY                  ; yes - exit
 042206   037 00 0 00 061624                    TMSGC   <Verify error at CRAM location >
 042207   200 00 0 00 037750                    MOVE    CADDR
 042210   037 04 0 00 000000                    PNT4
```

```
 042211   037 00 0 00 061633                    TMSGC   < Correct: >
```

```
042211  037 00 0 00 061633          TMSGC   < Correct: >
042212  200 02 0 00 064655'         MOVE    2,XWORDL                ; get left half
042213  200 03 0 00 064656'         MOVE    3,XWORDR                ; get right half
042214  260 17 0 00 043603          GO      PNTWD                   ; print it
042215  037 00 0 00 061636          TMSGC   < Actual:  >
042216  200 02 0 00 037751          MOVE    2,CWORDL                ; get left half
042217  200 03 0 00 037752          MOVE    3,CWORDR                ; get right half
042220  260 17 0 00 043603          GO      PNTWD                   ; print it
042221  037 00 0 00 030242          PCRL
042222  262 17 0 00 000003  DFVERY: RGET    (3,2,1,0)               ; restore AC's
042226  263 17 0 00 000000          RTN                             ; return



                            ;#*********************************************************************
                            ;* DFLIST - List microcode in file
                            ;#*********************************************************************

042227  200 01 0 00 061644  DFLIST: MOVE    1,[FMSGCD (FLIST <CR>)]
042230  260 17 0 00 042136*         GO      CHKARG                  ; check for argument
042231  263 17 0 00 000000          RTN                             ; error - exit
042232  260 17 0 00 042142*         GO      FINIT                   ; init file for reading
042233  263 17 0 00 000000           RTN                            ; error occurred - exit
042234  037 00 0 00 061076          TMSGC   <ADDR/ JUMP PAR OUT MGC SOR FUN DES CC RAM PA PB SK BUS CRY CTL  TM SP MK>
042235  260 17 0 00 042145* DFLISO: GO      FGETW                   ; get microcode entry
042236  254 00 0 00 042254           JRST   DFLISX                  ; error/EOF occurred
042237  200 02 0 00 037751          MOVE    2,CWORDL                ; get left half
042240  200 03 0 00 037752          MOVE    3,CWORDR                ; get right half
042241  037 00 0 00 030242          PCRL                            ; start a new line
042242  001 04 0 00 037750          PNTOCC  4,CADDR                 ; print CRAM load address
042243  037 00 0 00 061072          TMSG    </ >
042244  400 01 0 00 000000          SETZ    1,                      ; ignore 'bad parity' bit
042245  332 00 0 00 035764          SKIPE   PARFLG                  ; calculate parity?
042246  260 17 0 00 043625          GO      CALPAR                  ; yes - do so
042247  260 17 0 00 043507          GO      PNTCRM                  ; print it
042250  332 00 0 00 064570'         SKIPE   ALTF                    ; altmode typed?
042251  254 00 0 00 042254          JRST    DFLISX                  ; yes - exit
042252  037 07 0 00 000003          TTALTM                          ; altmode key struck?
042253  254 00 0 00 042235          JRST    DFLISO                  ; no - loop till done

042254  037 00 0 00 030242  DFLISX: PCRL                            ; final CRLF
042255  263 17 0 00 000000          RTN                             ; exit
```

```
                                        ;#**********************************************************************
                                        ;* DFEXAM - Examine microcode in file
                                        ;#**********************************************************************

042256  200 01 0 00 061650      DFEXAM: MOVE    1,[FMSGCD (FEXAM <CR>)]
042257  260 17 0 00 042230*             GO      CHKARG                  ; check for argument
042260  263 17 0 00 000000              RTN                             ; error - exit
042261  260 17 0 00 042232*             GO      FINIT                   ; init file for reading
042262  263 17 0 00 000000               RTN                            ; error occurred - exit
042263  260 17 0 00 042235*     DFEXA0: GO      FGETW                   ; get microcode entry
042264  254 00 0 00 042302                JRST   DFEXAX                  ; error/EOF occurred
042265  200 02 0 00 037751              MOVE    2,CWORDL                ; get left half
042266  200 03 0 00 037752              MOVE    3,CWORDR                ; get right half
042267  037 00 0 00 030242              PCRL                            ; start a new line
042270  001 04 0 00 037750              PNTOCC  4,CADDR                 ; print CRAM load address
042271  037 00 0 00 061072              TMSG    </ >
042272  400 01 0 00 000000              SETZ    1,                      ; ignore 'bad parity' bit
042273  332 00 0 00 035764              SKIPE   PARFLG                  ; calculate parity?
042274  260 17 0 00 043625              GO      CALPAR                  ; yes - do so
042275  260 17 0 00 043603              GO      PNTWD                   ; print it
042276  332 00 0 00 064570*             SKIPE   ALTF                    ; altmode typed?
042277  254 00 0 00 042302              JRST    DFEXAX                  ; yes - exit
042300  037 07 0 00 000003              TTALTM                          ; altmode key struck?
042301  254 00 0 00 042263              JRST    DFEXA0                  ; no - loop till done

042302  037 00 0 00 030242      DFEXAX: PCRL                            ; final CRLF
042303  263 17 0 00 000000              RTN                             ; exit


                                        ;#**********************************************************************
                                        ;* DTRANS - Translate CSR bits to English
                                        ;#**********************************************************************

042304  210 01 0 00 061656      DTRANS: MOVN    1,[FMSGCD (TRANSL data <CR>)]
042305  260 17 0 00 041770*             GO      .OARG                   ; get argument
042306  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
                                         JRST   [FMSGCD <? Missing argument>
                                                 GO      FIOFF
042307  254 00 0 00 055164                       RTN]
042310  200 01 0 00 042066*             MOVE    1,ARGUM                 ; get argument
042311  260 17 0 00 043715              GO      CSRPNT                  ; print CSR data in English
042312  037 00 0 00 030242              PCRL                            ; end of line
042313  263 17 0 00 000000              RTN                             ; return
```

```
                                        ;**********************************************************************
```

```

                                    ;#*******************************************************************
                                    ;* DCCODE - Print condition codes
                                    ;#*******************************************************************

042314  200 01 0 00 061662          DCCODE: MOVE    1,[FMSGCD (CCODE <CR>)]
042315  260 17 0 00 042257*                 GO      CHKARG                  ; check for argument
042316  263 17 0 00 000000                  RTN                             ; error - exit

                                    ; Save CRAM locations

042317  200 01 0 00 061663                  MOVE    1,[-4,,7750]            ; set up AOBJN word
042320  260 17 0 00 044040                  GO      SAVCRM                  ; save the CRAM locations

                                    ; Loop on condition code

042321  474 07 0 00 000000                  SETO    7,                      ; initialize pointer
042322  350 00 0 00 000007          DCCOD1: AOS     7                       ; point to next condition code
042323  303 07 0 00 000017                  CAILE   7,17                    ; done yet?
042324  254 00 0 00 042346                  JRST    DCCOD2                  ; yes - go print results
042325  137 07 0 00 061664                  DPB     7,[POINT 4,CCMIC+3,23]  ; no - set up microcode
042326  201 01 0 00 042432                  MOVEI   1,CCMIC                 ; set up microcode address
042327  260 17 0 00 043207                  GO      MLOADN                  ; load/verify it
042330  255 00 0 00 000000                  JFCL                            ; ignore error
042331  201 01 0 00 007750                  MOVEI   1,7750                  ; get start address
042332  242 01 0 00 000001                  LSH     1,1                     ; position correctly
042333  260 17 0 00 041204*                 GO      LDRAR                   ; load the RAR
042334  201 01 0 00 000010                  MOVEI   1,MPRUN                 ; start it up
042335  260 17 0 00 041210*                 GO      LDCSR                   ; write to CSR register
042336  201 01 0 00 040000                  MOVEI   1,SELLAR                ; set up to read LAR
042337  260 17 0 00 042335*                 GO      LDCSR                   ; stop the port
042340  260 17 0 00 037627*                 GO      RDLAR                   ; read stopping address
042341  242 01 0 00 777777                  LSH     1,-1                    ; position correctly
042342  402 00 0 07 042412                  SETZM   CCTAB3(7)               ; initialize result
042343  306 01 0 00 007753                  CAIN    1,7753                  ; was the cc set?
042344  476 00 0 07 042412                  SETOM   CCTAB3(7)               ; yes - indicate such
042345  254 00 0 00 042322                  JRST    DCCOD1                  ; loop till done

                                    ; Print the results

042346  037 00 0 00 061665          DCCOD2: TMSGC   <CC On:  >
042347  474 07 0 00 000000                  SETO    7,                      ; initialize pointer
042350  350 00 0 00 000007          DCCOD3: AOS     7                       ; point to next entry
042351  303 07 0 00 000017                  CAILE   7,17                    ; done yet?
042352  254 00 0 00 042356                  JRST    DCCOD4                  ; yes - continue
042353  332 00 0 07 042412                  SKIPE   CCTAB3(7)               ; no - cc on?
042354  256 00 0 07 042372                  XCT     CCTAB1(7)               ; yes - print text
042355  254 00 0 00 042350                  JRST    DCCOD3
```

```

042356  037 00 0 00 061670      DCCOD4: TMSGC   <CC Off: >
042357  474 07 0 00 000000              SETO    7,                      ; initialize pointer
042360  350 00 0 00 000007      DCCOD5: AOS     7                       ; point to next entry
042361  303 07 0 00 000017              CAILE   7,17                    ; done yet?
042362  254 00 0 00 042366              JRST    DCCOD6                  ; yes - continue
042363  336 00 0 07 042412              SKIPN   CCTAB3(7)               ; no - cc on?
042364  256 00 0 07 042372              XCT     CCTAB1(7)               ; yes - print text
042365  254 00 0 00 042360              JRST    DCCOD5
042366  037 00 0 00 030242      DCCOD6: PCRL


                                ; Restore CRAM locations and exit

042367  200 01 0 00 061663              MOVE    1,[-4,,7750]            ; set up AOBJN word
042370  260 17 0 00 044062              GO      RESCRM                  ; restore CRAM data
042371  263 17 0 00 000000              RTN                             ; return

                                ; Tables

042372  037 00 0 00 061673      CCTAB1: TMSG    < CBAV>                 ; 0 - CCCBUSAVAIL
042373  037 00 0 00 061675              TMSG    < GCSR>                 ; 1 - CCGRNTCSR
042374  037 00 0 00 061677              TMSG    < FEQ0>                 ; 2 - CCFEQ0
042375  037 00 0 00 061701              TMSG    < CSRC>                 ; 3 - CCCSRCHNG
042376  037 00 0 00 061703              TMSG    < EBPE>                 ; 4 - CCEBPARERR
042377  037 00 0 00 061705              TMSG    < AFUL>                 ; 5 - CCRCVRBUFAFUL
042400  037 00 0 00 061707              TMSG    < BFUL>                 ; 6 - CCRCVRBUFBFUL
042401  037 00 0 00 061711              TMSG    < XATN>                 ; 7 - CCXMTRATTN
042402  037 00 0 00 061713              TMSG    < ERQS>                 ; 10 - CCEBUSRQST
042403  037 00 0 00 061715              TMSG    < IACT>                 ; 11 - CCINTRACTIVE
042404  037 00 0 00 061717              TMSG    < MSGN>                 ; 12 - CCMBSIGN
042405  037 00 0 00 061721              TMSG    < MVRP>                 ; 13 - CCMVRPARCHK
042406  037 00 0 00 061723              TMSG    < CBPE>                 ; 14 - CCCBUSPARERR
042407  037 00 0 00 061725              TMSG    < PLPE>                 ; 15 - CCPLIPARERR
042410  037 00 0 00 061727              TMSG    < CHER>                 ; 16 - CCCHANERR
042411  037 00 0 00 061731              TMSG    < CBLW>                 ; 17 - CCCBLSTWD

042412                          CCTAB3: BLOCK   ^D16                    ; results

                                ; Microcode

042432  775077  510000          CCMIC:  MWORD   <ADDR=7750,JMAP,J=7751,D=1>     ; 7750
042434  775177  530000                  MWORD   <CJP,J=7753,D=1,CENA>           ; 7751
042436  775277  520000                  MWORD   <JMAP,J=7752,D=1>               ; 7752
042440  775377  530000                  MWORD   <JMAP,J=7753,D=1>               ; 7753
042442  777777  777777                  -1
```

```
                                           ;#*************************************************************************
                                           ;* DFILLN - Fill CRAM with special data (J=PC+1,OPCode)
                                           ;#*************************************************************************

042443  210 01 0 00 061743                 DFILLN: MOVN    1,[FMSGCD <FILLNX number (2910 Op code 0-17)>]
042444  260 17 0 00 042305*                        GO      .OARG                      ; get argument
042445  263 17 0 00 000000                          RTN                               ; error/altmode/question - exit
                                                     JRST   [FMSGCD <? Missing argument>
                                                             GO      FIOFF
042446  254 00 0 00 055164                                   RTN]
042447  260 17 0 00 042144*                        GO      ISTOP                      ; stop the port
042450  200 01 0 00 042310*                        MOVE    1,ARGUM                    ; get argument
042451  400 02 0 00 000000                         SETZ    2,                         ; initialize address
042452  402 00 0 00 037751                         SETZM   CWORDL                     ; initialize left half
042453  402 00 0 00 037752                         SETZM   CWORDR                     ; initialize right half
042454  202 02 0 00 037750                 DFILNO: MOVEM   2,CADDR                    ; set up address
042455  350 00 0 00 000002                         AOS     2                          ; point to next address
042456  303 02 0 00 007777                         CAILE   2,7777                     ; at end yet?
042457  400 02 0 00 000000                         SETZ    2,                         ; yes - set back to zero
042460  137 02 0 00 060366                         DPB     2,[POINT 12,CWORDL,17]     ; insert J field
042461  137 01 0 00 060440                         DPB     1,[POINT 4,CWORDR,31]      ; insert CTL field
042462  260 17 0 00 040376                         GO      DWCRAM                     ; load it
042463  326 02 0 00 042454                         JUMPN   2,DFILNO                   ; loop till done
042464  037 00 0 00 061744                         TMSGCD  <[Number of CRAM locations loaded = 4096.]>
042465  263 17 0 00 000000                         RTN                                ; return


                                           ;#*************************************************************************
                                           ;* DFILLP - Fill CRAM with special data (J=PC,OPCode)
                                           ;#*************************************************************************

042466  210 01 0 00 061766                 DFILLP: MOVN    1,[FMSGCD <FILLPC number (2910 Op code 0-17)>]
042467  260 17 0 00 042444*                        GO      .OARG                      ; get argument
042470  263 17 0 00 000000                          RTN                               ; error/altmode/question - exit
                                                     JRST   [FMSGCD <? Missing argument>
                                                             GO      FIOFF
042471  254 00 0 00 055164                                   RTN]
042472  260 17 0 00 042447*                        GO      ISTOP                      ; stop the port
042473  200 01 0 00 042450*                        MOVE    1,ARGUM                    ; get argument
042474  476 00 0 00 037750                         SETOM   CADDR                      ; initialize address
042475  402 00 0 00 037751                         SETZM   CWORDL                     ; initialize left half
042476  402 00 0 00 037752                         SETZM   CWORDR                     ; initialize right half
042477  350 02 0 00 037750                 DFILPO: AOS     2,CADDR                    ; set up address
042500  137 02 0 00 060366                         DPB     2,[POINT 12,CWORDL,17]     ; insert J field
042501  137 01 0 00 060440                         DPB     1,[POINT 4,CWORDR,31]      ; insert CTL field
042502  260 17 0 00 040376                         GO      DWCRAM                     ; load it
042503  307 02 0 00 007777                         CAIG    2,7777                     ; done yet?
042504  254 00 0 00 042477                         JRST    DFILPO                     ; no - loop till done
042505  037 00 0 00 061744                         TMSGCD  <[Number of CRAM locations loaded = 4096.]>
042506  263 17 0 00 000000                         RTN                                ; return
```

```
                                        ;#***********************************************************************
                                        ;* DRDINI - Initialize everything for a read
                                        ;#***********************************************************************

042507  200 01 0 00 061773      DRDINI: MOVE    1,[FMSGCD (RDINIT <CR>)]
042510  260 17 0 00 042315*             GO      CHKARG                          ; check for argument
042511  263 17 0 00 000000              RTN                                     ; error - exit
042512  201 01 0 00 000001              MOVEI   1,1                             ; set 'read' flag

042513  202 01 0 00 042531      DRDINC: MOVEM   1,LASDTC                        ; save operation
042514  200 14 0 00 035767              MOVE    PAT,DSPAT                       ; get buffer pattern
042515  200 00 0 00 035770              MOVE    DSTPAT                          ; get operator select pattern
042516  202 00 0 00 052002              MOVEM   TPAT                            ; save it
042517  200 02 0 00 035766              MOVE    2,DSWORD                        ; get buffer length
042520  202 02 0 00 052000              MOVEM   2,TWORD                         ; save it also
042521  260 17 0 00 000000*             GO      BUFGEN                          ; insert buffer pattern
042522  200 01 0 00 061774              MOVE    1,[CBUFSZ,,CBUF]                ; get pointer to control buffer
042523  260 17 0 00 000000*             GO      CHINIT                          ; initialize software
042524  551 01 0 00 050000              HRRZI   1,BUFF                          ; buffer address
042525  200 02 0 00 052000              MOVE    2,TWORD                         ; word count
042526  400 03 0 00 000000              SETZ    3,                              ; clear zero fill specifier
042527  260 17 0 00 000000*             GO      GENCCW                          ; generate a CCW list
042530  263 17 0 00 000000              RTN                                     ; return

042531  777777  777777          LASDTC: -1                                      ; last data transfer operation



                                        ;#***********************************************************************
                                        ;* DWRINI - Initialize everything for a write
                                        ;#***********************************************************************

042532  200 01 0 00 062001      DWRINI: MOVE    1,[FMSGCD (WRINIT <CR>)]
042533  260 17 0 00 042510*             GO      CHKARG                          ; check for argument
042534  263 17 0 00 000000              RTN                                     ; error - exit
042535  201 01 0 00 000002              MOVEI   1,2                             ; set 'write' flag
042536  254 00 0 00 042513              JRST    DRDINC                          ; continue at common code
```

```
                                        ***************************
```

```
                                          ;#**********************************************************************
                                          ;* DBPNT - Print buffer contents
                                          ;#**********************************************************************

042537  200 00 0 00 035766   DBPNT:  MOVE    DSWORD                          ; set default number to length
042540  303 00 0 00 000024           CAILE   ^D20                            ;   of buffer or 20 words
042541  201 00 0 00 000024           MOVEI   ^D20                            ;   whichever is smaller
042542  202 00 0 00 042473*          MOVEM   ARGUM
042543  210 01 0 00 062012           MOVN    1,[FMSGCD (BPRINT WordCount or BPRINT <CR>)]
042544  260 17 0 00 041460*          GO      .DARG                           ; get argument
042545  263 17 0 00 000000            RTN                                    ; error/altmode/question - exit
042546  255 00 0 00 000000            JFCL                                   ; no argument given
042547  210 03 0 00 042542*          MOVN    3,ARGUM                         ; get argument
042550  514 03 0 00 000003           HRLZ    3,3                             ; make AOBJN pointer
042551  541 03 0 00 050000           HRRI    3,BUFF                          ; insert buffer address
042552  550 00 0 00 000003   DBPNT0: HRRZ    3                               ; get address
042553  275 00 0 00 050000           SUBI    BUFF                            ; normalize to 0..n-1
042554  602 00 0 00 000003           TRNE    3                               ; multiple of 4?
042555  037 00 0 00 055371           TMSG    <  >                            ; no - 2 spaces
042556  602 00 0 00 000003           TRNE    3                               ; multiple of 4?
042557  254 00 0 00 042564           JRST    DBPNT1                          ; no - don't print address
042560  037 00 0 00 062013           TMSGC   <Wrd >                          ; print word location
042561  350 00 0 00 000000           AOS                                     ; make from 1..n
042562  037 15 0 00 000000           PNTDEC
042563  037 00 0 00 062015           TMSG    <, >
042564  200 00 0 03 000000   DBPNT1: MOVE    (3)                             ; get data
042565  037 13 0 00 000000           PNTHW                                   ; print data
042566  037 07 0 00 000003           TTALTM                                  ; altmode typed?
042567  253 03 0 00 042552           AOBJN   3,DBPNT0                        ; no - do next word - loop till done
042570  037 00 0 00 030242           PCRL                                    ; final CRLF
042571  263 17 0 00 000000           RTN                                     ; return


                                          ;#**********************************************************************
                                          ;* DCWPNT - Print CCW list
                                          ;#**********************************************************************

042572  200 01 0 00 062022   DCWPNT: MOVE    1,[FMSGCD (CCWPNT <CR>)]
042573  260 17 0 00 042533*          GO      CHKARG                          ; check for argument
042574  263 17 0 00 000000           RTN                                     ; error - exit
042575  260 17 0 00 000000*          GO      CCWPNT                          ; print CCW list
042576  037 00 0 00 030242           PCRL
042577  263 17 0 00 000000           RTN                                     ; return
```

```
 6688
 6689                                        ;#*****************************************************************
 6690                                        ;* DLGPNT - Print logout data
 6691                                        ;#*****************************************************************
 6692
 6693   042600  200 01 0 00 062027           DLGPNT: MOVE    1,[FMSGCD (LOGPNT <CR>)]
 6694   042601  260 17 0 00 042573*                  GO      CHKARG                  ; check for argument
 6695   042602  263 17 0 00 000000                   RTN                             ; error - exit
 6696   042603  201 01 0 00 000000*                  MOVEI   1,LOGBUF                ; set up logout data storage area
 6697   042604  260 17 0 00 000000*                  GO      GETLOG                  ; get logout data to a buffer area
 6698   042605  260 17 0 00 000000*                  GO      LOGPNT                  ; print channel logout data
 6699   042606  037 00 0 00 030242                   PCRL
 6700   042607  263 17 0 00 000000                   RTN                             ; return
 6701
 6702                                        ;#*****************************************************************
 6703                                        ;* DCOMPA - Compare data buffer
 6704                                        ;#*****************************************************************
 6705
 6706
 6707   042610  200 01 0 00 062034           DCOMPA: MOVE    1,[FMSGCD (COMPAR <CR>)]
 6708   042611  260 17 0 00 042601*                  GO      CHKARG                  ; check for argument
 6709   042612  263 17 0 00 000000                   RTN                             ; error - exit
 6710   042613  200 01 0 00 042531                   MOVE    1,LASDTC                ; get last data transfer setup cmd
 6711   042614  301 01 0 00 000001                   CAIL    1,1                     ; this command valid?
 6712   042615  303 01 0 00 000003                   CAILE   1,3
 6713                                                JRST    [FMSGCD <? No data transfer setup command executed yet>
 6714                                                         GO      FIOFF
 6715   042616  254 00 0 00 062047                            RTN]
 6716   042617  200 02 0 00 035766                   MOVE    2,DSWORD                ; get buffer length
 6717   042620  260 17 0 00 000000*                  GO      BUFCOM                  ; do the buffer compare
 6718   042621  332 00 0 00 000000*                  SKIPE   WRDERR                  ; any errors?
 6719   042622  037 00 0 00 030242                   PCRL                            ; yes - end with a CRLF
 6720   042623  263 17 0 00 000000                   RTN                             ; return
 6721
 6722                                        ;#*****************************************************************
 6723                                        ;* DSWIT - Switch complement commands
 6724                                        ;#*****************************************************************
 6725
 6726
 6727   042624  200 01 0 00 055220           DSWIT:  MOVE    1,[FMSGCD (Switch Name <CR>)]
 6728   042625  260 17 0 00 042611*                  GO      CHKARG                  ; check for argument
 6729   042626  263 17 0 00 000000                   RTN                             ; error - exit
 6730   042627  261 17 0 00 000001                   PUT     1                       ; save AC1
 6731   042630  200 01 0 00 032707*                  MOVE    1,.CGOT                 ; get dispatch number
 6732   042631  275 01 0 00 000116                   SUBI    1,SWSTP                 ; normalize to 0..
 6733   042632  260 17 0 00 032711*                  GO      SWCOM                   ; complement switch
 6734   042633  262 17 0 00 000001                   GET     1                       ; restore AC1
 6735   042634  263 17 0 00 000000                   RTN                             ; continue
```

```
                                        ;#*****************************************************************
                                        ;* IPRINT - Print interrupt history
                                        ;#*****************************************************************

042635  200 01 0 00 062056              DIPRIN: MOVE    1,[FMSGCD (IPRINT <CR>)]
042636  260 17 0 00 042625*                     GO      CHKARG                  ; check for argument
042637  263 17 0 00 000000                      RTN                             ; error - exit
042640  261 17 0 00 000000              IPRINT: RPUT    (0,1,2)                 ; save AC's
042643  335 01 0 00 000000*             IPRINO: SKIPGE  1,INTNUM                ; any interrupt yet?
042644  254 00 0 00 042650                      JRST    IPRINX
042645  260 17 0 00 042654                      GO      IPRINI                  ; yes - print the data
042646  037 07 0 00 000003                      TTALTM                          ; check for altmode typed
042647  254 00 0 00 042643                      JRST    IPRINO                  ; no - keep looping
042650  262 17 0 00 000002              IPRINX: RGET    (2,1,0)                 ; yes - restore AC's
042653  263 17 0 00 000000                      RTN                             ; exit

                                        ; Print interrupt data

042654  370 01 0 00 042643*             IPRINI: SOS     1,INTNUM                ; point to previous interrupt
042655  303 01 0 00 000023                      CAILE   1,^D19                  ; overflow interrupt?
042656  201 01 0 00 000023                      MOVEI   1,^D19                  ; yes - limit to 20 interrupts
042657  202 01 0 00 042654*                     MOVEM   1,INTNUM                ; save interrupt number
042660  037 00 0 00 062057                      TMSGC   <=====>
042661  550 02 0 01 000000*                     HRRZ    2,INTTYP(1)             ; get type
                                                XCT     [TMSGC  <Vectored (PC = >
                                                         TMSGC  <NV Chn 1 (PC = >
                                                         TMSGC  <NV Chn 2 (PC = >
                                                         TMSGC  <NV Chn 3 (PC = >
                                                         TMSGC  <NV Chn 4 (PC = >
                                                         TMSGC  <NV Chn 5 (PC = >
                                                         TMSGC  <NV Chn 6 (PC = >
042662  256 00 0 02 062121                               TMSGC  <NV Chn 7 (PC = >](2)
042663  550 00 0 01 000000*                     HRRZ    INTPC(1)                ; get interrupt PC
042664  037 16 0 00 000003                      PNTOCS                          ; print it
042665  037 00 0 00 062131                      TMSG    <) at >
042666  200 00 0 01 000000*                     MOVE    INTTIM(1)               ; get milliseconds
042667  037 15 0 00 000000                      PNTDEC
042670  037 00 0 00 062133                      TMSG    <.ms + >
042671  200 00 0 01 000000*                     MOVE    INTUSE(1)               ; get microseconds
042672  037 15 0 00 000000                      PNTDEC
042673  037 00 0 00 062135                      TMSG    <.us>
042674  037 00 0 00 062136                      TMSGC   <CONI PI = >
042675  200 00 0 01 000000*                     MOVE    INTCON(1)               ; get CONI PI data
042676  037 13 0 00 000000                      PNTHW                           ; print it
042677  037 00 0 00 062141                      TMSG    <  CONI APR = >
042700  200 00 0 01 000000*                     MOVE    INTAPR(1)               ; get APR data
042701  037 13 0 00 000000                      PNTHW                           ; print it
```

```
042702  037 00 0 00 062144                      TMSG    <  CSR = >
```

```
042702  037 00 0 00 062144           TMSG    <  CSR = >
042703  200 00 0 01 000000*          MOVE    INTCSR(1)               ; get CSR data
042704  037 13 0 00 000000           PNTHW                           ; print it
042705  037 00 0 00 030242           PCRL
042706  335 00 0 01 042661*          SKIPGE  INTTYP(1)
042707  037 00 0 00 062146           TMSGCD  < ##### (Interrupts flushed) #####>
042710  263 17 0 00 000000           RTN



                            ;*********************************************************************
                            ;* IINIT - Initialize interrupt system
                            ;*********************************************************************

042711  200 01 0 00 062161  DIINIT: MOVE    1,[FMSGCD (IINIT <CR>)]
042712  260 17 0 00 042636*         GO      CHKARG                  ; check for argument
042713  263 17 0 00 000000          RTN                             ; error - exit
042714  260 17 0 00 030775*         GO      INITPI                  ; init PI system
042715  260 17 0 00 032300*         GO      SETVEC                  ; set up vector addresses
042716  263 17 0 00 000000          RTN                             ; return
```

```
                                     SUBTTL  Program Utility Routines
```

```
                                                   SUBTTL  Program Utility Routines

                                        ;#*********************************************************************
                                        ;* TRACE - Program Trace Routine
                                        ;
                                        ; This routine traces test execution if the TRACE switch is set.  This
                                        ; routine sets up TSTSUB, ERFLG, ERRPC for error reporting during  the
                                        ; test, and ERRNUM for interrupt handling.
                                        ;#*********************************************************************

042717  261 17 0 00 000000              TRACE:  RPUT    (0,1)                           ; save AC's
042721  202 00 0 00 047026                      MOVEM   TSTOFF                          ; save relocatable offset
042722  260 17 0 00 034410*                     GO      SWITT                           ; get switches
042723  606 00 0 00 400000                      TRNN    TRACET                          ; want to trace?
042724  254 00 0 00 042743                      JRST    TRACE0                          ; no

                                        ; Print test number if tracing

042725  037 00 0 00 062162                      TMSG    <* >
042726  260 17 0 00 033310*                     GO      PNTDEV                          ; print device name
042727  037 00 0 00 062163                      TMSG    < Test >
042730  200 01 0 00 047010                      MOVE    1,TSTCLS                        ; get test class
042731  200 00 0 01 031275                      MOVE    INDLIS(1)                       ; get class description
042732  037 00 0 00 000002                      PNTSIX                                  ; print it
042733  200 00 0 00 047011                      MOVE    TSTNUM                          ; get test number
042734  037 16 0 00 000003                      PNTOCS                                  ; print it
042735  200 00 0 00 047014                      MOVE    TSTFLG                          ; get test flags
042736  603 00 0 00 000100                      TLNE    (TDENA)                         ; disabled?
042737  037 00 0 00 062165                      TMSG    < (Disabled)>                   ; yes - print such
042740  603 00 0 00 001000                      TLNE    (TUSER)                         ; run in user mode?
042741  037 00 0 00 062170                      TMSG    < (Exec only test)>             ; no - print such
042742  037 00 0 00 030242                      PCRL

                                        ; Initialize flags

042743  402 00 0 00 000015              TRACE0: SETZM   ERFLG                           ; clear the error detected flag
042744  402 00 0 00 030052                      SETZM   ERRPC                           ; clear the error PC
042745  402 00 0 00 047012                      SETZM   TSTSUB                          ; clear subtest number
042746  402 00 0 00 000000*                     SETZM   SCOOFF                          ; clear error offset
042747  476 00 0 00 042657*                     SETOM   INTNUM                          ; initialize interrupt number
042750  402 00 0 00 041623*                     SETZM   SNEXT                           ; initialize next address
042751  402 00 0 00 041613*                     SETZM   SDATA                           ; initialize CSR data
```

```

                                        ; Determine whether or not test should be run

042752  200 00 0 00 047014                      MOVE    TSTFLG                          ; get test flags
042753  603 00 0 00 001100                      TLNE    (TDENA!TUSER)                   ; test disabled or exec mode only?
042754  254 00 0 00 042771                      JRST    TRACEX                          ; yes - exit test

                                        ; Handle microcode loaded flags

042755  316 16 0 00 047001                      CAMN    MBCN,PORTNI                     ; NI port selected?
042756  332 00 0 00 047016                      SKIPE   TSTMIC                          ; yes - any test microcode?
042757  334 00 0 00 000000                      SKIPA                                   ; no - continue
042760  476 00 0 00 064646'                     SETOM   TSLOD1                          ; yes - init ucode loaded flag (NI)
042761  316 16 0 00 047002                      CAMN    MBCN,PORTCI                     ; CI port selected?
042762  332 00 0 00 047016                      SKIPE   TSTMIC                          ; yes - any test microcode?
042763  334 00 0 00 000000                      SKIPA                                   ; no - continue
042764  476 00 0 00 064647'                     SETOM   TSLOD2                          ; yes - init ucode loaded flag (CI)

                                        ; Exit and run test

042765  262 17 0 00 000001                      RGET    (1,0)                           ; restore AC's
042767  350 00 0 17 777777                      AOS     -1(P)                           ; increment test return address
042770  263 17 0 00 000000                      RTN                                     ; exit normally

                                        ; Exit and skip test

042771  262 17 0 00 000001              TRACEX: RGET    (1,0,XXX)                       ; restore AC's
042774  263 17 0 00 000000                      RTN                                     ; exit test
```

```
                                            ;#*******************************************************************
                                            ;* TSTGET - Get test related data based on test dispatch address
                                            ;
                                            ; Given:  TSTADD - Contains address within a test dispatch table, this is
                                            ;                  location contains the address of the current test.
                                            ;
                                            ; Returns:  TSTCLS - Test class 0-5
                                            ;           TSTNUM - Test number
                                            ;           TSTSUB - Zeroed test offset
                                            ;           TSTNAM - Address of ASCIZ test description
                                            ;           TSTPC -  Current test PC
                                            ;           TSTFLG - Current test flags
                                            ;           TSTADD - Current test descriptor entry address
                                            ;           TSTMIC - Current test microcode address
                                            ;           TSTHIG - Addr of high probability modules
                                            ;           TSTLOW - Addr of low probability modules
                                            ;
                                            ;#*******************************************************************

042775  261 17 0 00 000001          TSTGET: RPUT    (1,2)                   ; save AC's
042777  402 00 0 00 047012                  SETZM   TSTSUB                  ; clear subtest number
043000  200 01 1 00 047015                  MOVE    1,@TSTADD               ; get test address
043001  202 01 0 00 047013                  MOVEM   1,TSTPC                 ; save it
043002  200 02 0 01 000001                  MOVE    2,1(1)                  ; get test flags
043003  202 02 0 00 047014                  MOVEM   2,TSTFLG                ; save them
043004  554 02 0 01 000002                  HLRZ    2,2(1)                  ; get microcode address
043005  202 02 0 00 047016                  MOVEM   2,TSTMIC                ; save it
043006  550 02 0 01 000002                  HRRZ    2,2(1)                  ; get test description
043007  202 02 0 00 047007                  MOVEM   2,TSTNAM                ; save it
043010  554 02 0 01 000003                  HLRZ    2,3(1)                  ; get address of high prob modules
043011  202 02 0 00 047024                  MOVEM   2,TSTHIG                ; save it
043012  550 02 0 01 000003                  HRRZ    2,3(1)                  ; get address of low prob modules
043013  202 02 0 00 047025                  MOVEM   2,TSTLOW                ; save it
043014  550 01 0 00 047014                  HRRZ    1,TSTFLG                ; get test number
043015  202 01 0 00 047011                  MOVEM   1,TSTNUM                ; save it
043016  135 01 0 00 062174                  LDB     1,[POINT 3,TSTFLG,17]   ; get test class
043017  202 01 0 00 047010                  MOVEM   1,TSTCLS                ; save it
043020  262 17 0 00 000002                  RGET    (2,1)                   ; restore AC's
043022  263 17 0 00 000000                  RTN
```

```
                                            ;#*******************************************************************
```

```
                                        ;#**********************************************************************
                                        ;* SELTST - Select a test to run.
                                        ;
                                        ; This routine selects the next test in a test descriptor  table.
                                        ; TSTADD contains the address within a test dispatch table of the
                                        ; current test to run.  If there are no more tests  to  run, this
                                        ; will contain -1.
                                        ;
                                        ; Return:        +1 - No more tests found
                                        ;                +2 - Test found is not valid
                                        ;                +3 - Test found is valid
                                        ;
                                        ;#**********************************************************************

043023  261 17 0 00 000000              SELTST: RPUT    (0,1,2)                 ; save AC's
043026  350 01 0 00 047015                      AOS     1,TSTADD                ; get address in dispatch table
043027  200 02 0 01 000000                      MOVE    2,(1)                   ; get dispatch address
043030  316 02 0 00 060261                      CAMN    2,[-1]                  ; end of tests?
043031  254 00 0 00 043037                      JRST    SELTSX                  ; yes - exit +1

                                        ; Obtain test data

043032  260 17 0 00 042775                      GO      TSTGET                  ; get test data
043033  260 17 0 00 043043                      GO      CHKTST                  ; check if test is valid
043034  334 00 0 00 000000                      SKIPA                           ; no - exit +2 (test not valid)
043035  350 00 0 17 777775                      AOS     -3(P)                   ; set up exit +3 (Found a test)
043036  350 00 0 17 777775                      AOS     -3(P)                   ; set up exit +2 (Test not valid)

                                        ; Exit

043037  262 17 0 00 000002              SELTSX: RGET    (2,1,0)                 ; restore AC's
043042  263 17 0 00 000000                      RTN                             ; return
```

```
                                        ;#**********************************************************************
```

```
                                  ;#*******************************************************************
                                  ;* CHKTST - Check if this is a valid test to execute and set up test type.
                                  ;
                                  ; A test is determined to be a valid test if the following are true:
                                  ;
                                  ;  - A class of tests has been selected, and this test is in that class.
                                  ;  - The test can be run, ie., that the modules specified as required
                                  ;    are not indicated as missing by the switches.
                                  ;
                                  ; Argument:     TSTPC - Contains test address
                                  ;
                                  ; Return:       +1 - Not a valid test
                                  ;               +2 - Valid test
                                  ;#*******************************************************************

 043043  261 17 0 00 000000       CHKTST: RPUT    (0,1,2)                        ; save some AC's
 043046  200 01 0 00 047013               MOVE    1,TSTPC                        ; get test pc
 043047  331 00 0 00 047022               SKIPL   TSTMUL                         ; multiple tests?
 043050  254 00 0 00 043054               JRST    CHKTS0                         ; no - continue

                                  ; Verify proper class of tests

 043051  200 02 0 00 047017               MOVE    2,TSTMSK                       ; get test mask
 043052  616 02 0 01 000001               TDNN    2,1(1)                         ; valid test to execute?
 043053  254 00 0 00 043072               JRST    CHKTSX                         ; no - exit (+1)

                                  ; Check if there are enough boards here to run this

 043054  400 00 0 00 000000       CHKTS0: SETZ    0,                             ; clear switches
 043055  332 00 0 00 030056               SKIPE   $ONETM                         ; initial startup?
 043056  260 17 0 00 042722*              GO      SWITT                          ; no - get program switches
 043057  606 00 0 00 000003               TRNN    MMPROC!MCBUS                   ; any modules missing?
 043060  254 00 0 00 043071               JRST    CHKTS1                         ; no - continue
 043061  602 00 0 00 000002               TRNE    MMPROC                         ; MPROC module missing?
 043062  607 02 0 00 000600               TLNN    2,(NDMP!NDCB)                  ; need this module?
 043063  334 00 0 00 000000               SKIPA                                  ; no - check CBUS module
 043064  254 00 0 00 043072               JRST    CHKTSX                         ; yes - exit (+1) (not a valid test)
 043065  602 00 0 00 000001               TRNE    MCBUS                          ; CBUS module missing?
 043066  607 02 0 00 000200               TLNN    2,(NDCB)                       ; need this module?
 043067  334 00 0 00 000000               SKIPA                                  ; no - continue (valid test)
 043070  254 00 0 00 043072               JRST    CHKTSX                         ; yes - exit (+1) (not a valid test)
 043071  350 00 0 17 777775       CHKTS1: AOS     -3(P)                          ; set up RTN +2
 043072  262 17 0 00 000002       CHKTSX: RGLT    (2,1,0)                        ; restore AC's
 043075  263 17 0 00 000000               RTN                                    ; return
```

```
                                  ;#*******************************************************************
```

```
                                              ;#********************************************************************
                                              ;* TSTPNT - Print test name/number
                                              ;#********************************************************************

043076  261 17 0 00 000000      TSTPNT: RPUT    (0,1)                          ; save AC's
043100  200 01 0 00 047010              MOVE    1,TSTCLS                       ; get test class
043101  200 00 0 01 031275              MOVE    INDLIS(1)                      ; get class description
043102  037 00 0 00 000002              PNTSIX                                 ; print it
043103  200 00 0 00 047011              MOVE    TSTNUM                         ; get test number
043104  037 16 0 00 000003              PNTOCS                                 ; print it
043105  200 00 0 00 047012              MOVE    TSTSUB                         ; get subtest number
043106  322 00 0 00 043111              JUMPE   TSTPNO                         ; zero?  yes - continue
043107  037 00 0 00 054201              TMSG    <->                            ; no - print it
043110  037 16 0 00 000003              PNTOCS
043111  260 17 0 00 043056*     TSTPNO: GO      SWITT                          ; get switches
043112  603 00 0 00 000200              TLNE    TXTINH                         ; text inhibit set?
043113  254 00 0 00 043116              JRST    TSTPNX                         ; yes - exit
043114  037 00 0 00 054243              TMSG    < - >
043115  037 00 1 00 047007              PNTMSG  @TSTNAM                        ; print test description
043116  262 17 0 00 000001      TSTPNX: RGET    (1,0)                          ; restore AC's
043120  263 17 0 00 000000              RTN
```

```
                                              ;#********************************************************************
```

```
                                       ;#**********************************************************************
                                       ;* ODELAY - Delay a specified amount of time (in milliseconds in AC0)
                                       ;#**********************************************************************

043121  332 00 0 00 030037             ODELAY: SKIPE   USER                        ; user mode?
043122  254 00 0 00 043140                     JRST    ODELA1                      ; yes - go handle

                                       ; Exec mode - inspect timers

043123  261 17 0 00 000002                     RPUT    (2,3)                       ; save some AC's
043125  400 03 0 00 000000                     SETZ    3,                          ; set up so RUNTME doesn't print time
043126  260 17 0 00 033323*                    GO      RUNTME                      ; get initial time
043127  200 02 0 00 000000*                    MOVE    2,DEBTIM                    ; get time in msecs
043130  272 00 0 00 000002                     ADDM    2                           ; add in the time to delay
043131  400 03 0 00 000000             ODELA0: SETZ    3,                          ; set up so RUNTME doesn't print time
043132  260 17 0 00 043126*                    GO      RUNTME                      ; get time
043133  313 02 0 00 043127*                    CAMLE   2,DEBTIM                    ; done?
043134  254 00 0 00 043131                     JRST    ODELA0                      ; no - keep checking time
043135  262 17 0 00 000003                     RGET    (3,2)                       ; restore AC's
043137  263 17 0 00 000000                     RTN                                 ; return

                                       ; User mode - use DISMS JSYS or HIBER UUO

043140  261 17 0 00 000000             ODELA1: RPUT    (0,1)                       ; save AC's
043142  336 00 0 00 030516                     SKIPN   MONTYP                      ; TOPS-10?
043143  332 00 0 00 000000                     SKIPE   0                           ; yes, wait time of zero?
043144  334 01 0 00 000000                     SKIPA   1,0                         ; get time to wait
043145  201 01 0 00 000001                     MOVEI   1,1                         ; TOPS-10 and zero wait, make it 1 ms
043146  332 00 0 00 030516                     SKIPE   MONTYP                      ; TOPS-20?
043147  104 00 0 00 000167                     DISMS                               ; wait
043150  336 00 0 00 030516                     SKIPN   MONTYP                      ; TOPS-10?
043151  047 01 0 00 000072                     HIBER   1,                          ; wait
043152  255 00 0 00 000000                       JFCL                              ; HIBER UUO skip returns
043153  262 17 0 00 000001                     RGET    (1,0)                       ; restore AC's
043155  263 17 0 00 000000                     RTN                                 ; return
```

```
                                       ;#**********************************************************************
                                       ;#*********************************************************************
                                       ;* DDELAY - Delay a specified amount of time (in milliseconds in AC0)
                                       ;
                                       ; While waiting the data buffer is read and the CSR register is read to
                                       ; see if the test has signalled end of test by setting the ENABLE bit.
                                       ;#*********************************************************************

043156  261 17 0 00 000001      DDELAY: RPUT    (1,2,3,4)               ; save some AC's
043162  403 03 0 00 000004              SETZB   3,4                     ; set up so RUNTME doesn't print time
043163  260 17 0 00 043132*             GO      RUNTME                  ; get initial time
043164  200 02 0 00 043133*             MOVE    2,DEBTIM                ; get time in msecs
043165  272 00 0 00 000002              ADDM    2                       ; add in the time to delay
043166  400 03 0 00 000000      DDELA0: SETZ    3,                      ; set up so RUNTME doesn't print time
043167  260 17 0 00 043163*             GO      RUNTME                  ; get time
043170  317 02 0 00 043164*             CAMG    2,DEBTIM                ; done?
043171  254 00 0 00 043202              JRST    DDELAX                  ; yes - exit
043172  200 01 0 04 050000              MOVE    1,BUFF(4)               ; read a word in buffer
043173  350 00 0 00 000004              AOS     4                       ; point to next word
043174  301 04 0 00 001000              CAIL    4,^D512                 ; reach end of buffer?
043175  400 04 0 00 000000              SETZ    4,                      ; yes - reinitialize pointer
043176  260 17 0 00 036153*             GO      RDCSR                   ; read CSR
043177  255 00 0 00 000000              JFCL                            ; error
043200  606 01 0 00 000200              TRNN    1,RESQAV                ; RESQAV bit set?
043201  254 00 0 00 043166              JRST    DDELA0                  ; no - keep checking time
043202  262 17 0 00 000004      DDELAX: RGET    (4,3,2,1)               ; restore AC's
043206  263 17 0 00 000000              RTN                             ; return
```

```
                                       ;#*********************************************************************
```

```
                                         ;#*****************************************************************
                                         ;* MLOADN - Load microcode and verify it (do not print errors)
                                         ;* MLOADY - Load microcode and verify it (print errors)
                                         ;
                                         ; Arguments:    AC1 - Contains address where microcode starts
                                         ;
                                         ; Returns:      AC1 - Number of words loaded/verified
                                         ;               +1 - Error occurred loading/verifying microcode
                                         ;               +2 - Ok
                                         ;
                                         ; Microcode format:  The microcode is given in  2  word  segments
                                         ;       (specifying address and 60 CRAM bits).  The ucode is in a
                                         ;       table beginning at the address given in AC1.  The end  of
                                         ;       this table is indicated by a word containing -1.
                                         ;
                                         ;  Word 1:  Bits 00-11 - Microword load address (0000-7777)
                                         ;                12-35 - Bits 0-23 of microword to load
                                         ;  Word 2:  Bits 00-34 - Bits 24-57,59 of microword to load
                                         ;                35 ---- Force Bad Parity flag
                                         ;#*****************************************************************

  043207  402 00 0 00 064622'            MLOADN: SETZM   MVREP#                  ; clear print flag
  043210  334 00 0 00 000000                     SKIPA                           ; continue
  043211  476 00 0 00 064622'            MLOADY: SETOM   MVREP                   ; set print flag
  043212  402 00 0 00 064620'                    SETZM   MVFLAG#                 ; clear verify flag
  043213  261 17 0 00 000001                     PUT     1                       ; save AC1
  043214  260 17 0 00 043224                     GO      MVCOM                   ; load microcode
  043215  255 00 0 00 000000                     JFCL                            ; error - ignore
  043216  262 17 0 00 000001                     GET     1                       ; restore AC1
  043217  476 00 0 00 064620'                    SETOM   MVFLAG                  ; set verify flag
  043220  260 17 0 00 043224                     GO      MVCOM                   ; verify microcode
  043221  263 17 0 00 000000                     RTN                             ; error - return
  043222  350 00 0 17 000000                     AOS     (P)                     ; set up proper return
  043223  263 17 0 00 000000                     RTN                             ; return

  043224  476 00 0 00 064621'            MVCOM:  SETOM   MVNUM#                  ; init number of verify errors
                                                 JUMPLE  1,[SETZ 1,              ; exit if no microcode table specified
  043225  323 01 0 00 062175                                RTN]
  043226  261 17 0 00 000000                     RPUT    (0,2,3,4,5)             ; save some AC's
  043233  200 04 0 00 000001                     MOVE    4,1                     ; get address of table in AC4
  043234  200 01 0 00 062177                     MOVE    1,[MWBADR,,MWBADR+1]    ; build a BLT pointer
  043235  476 00 0 00 043340                     SETOM   MWBADR                  ; initialize first word
  043236  251 01 0 00 043356                     BLT     1,MWBRAC+2              ; initialize the rest

                                         ; First stop the port and ensure 'DIAG TEST EBUF' is not set

  043237  260 17 0 00 042472*                    GO      ISTOP                   ; stop the port
  043240  260 17 0 00 000000*                    GO      CLREBU                  ; ensure cleared
  043241  255 00 0 00 000000                      JFCL                           ; ignore error
```

```

                                         ; Load microwords
```

```
                                          ; Load microwords

043242  400 05 0 00 000000                         SETZ    5,                      ; clear word count
043243  200 02 0 04 000000                MLOAD0:  MOVE    2,(4)                   ; get first word
043244  316 02 0 00 060261                         CAMN    2,[-1]                  ; end of list?
043245  254 00 0 00 043325                         JRST    MLOADX                  ; yes - exit
043246  200 03 0 04 000001                         MOVE    3,1(4)                  ; get second word

                                          ; Load bits 0-27

043247  135 01 0 00 062200                         LDB     1,[POINT 12,2,11]       ; get actual CRAM load address
043250  202 01 0 00 037750                         MOVEM   1,CADDR                 ; save address
043251  242 01 0 00 000001                         LSH     1,1                     ; convert to address for RAR
043252  660 01 0 00 000001                         TRO     1,1                     ; set bit 12 (for left 30 bits)
043253  202 01 0 00 064640'                        MOVEM   1,SAVRAR#               ; save address loaded
043254  260 17 0 00 042333*                        GO      LDRAR                   ; load RAR with it
043255  200 01 0 00 000003                         MOVE    1,3                     ; get word 2 (it has 'bad parity' flag)
043256  135 02 0 00 062201                         LDB     2,[POINT 24,2,35]       ; get CRAM bits 0-23
043257  246 02 0 00 000006                         LSHC    2,6                     ; now get bits 0-29 in AC2 (left half)
043260  242 03 0 00 777772                         LSH     3,-6                    ; right justify AC3 (right half)
043261  622 03 0 00 000002                         TRZE    3,2                     ; bit 59 set?
043262  660 03 0 00 000001                         TRO     3,1                     ; yes - set rightmost bit
043263  260 17 0 00 043625                         GO      CALPAR                  ; calculate parity
043264  200 01 0 00 000002                         MOVE    1,2                     ; get left 30 bits
043265  336 00 0 00 064620'                        SKIPN   MVFLAG                  ; load?
043266  260 17 0 00 040422*                        GO      LDCRAM                  ; yes - load it
043267  332 00 0 00 064620'                        SKIPE   MVFLAG                  ; verify?
                                                   GO      [GO      RDCRAM         ; yes - read CRAM
                                                            MOVEM   1,CWORDL       ; save it
043270  260 17 0 00 062202                                  RTN]

                                          ; Load bits 30-59

043271  200 01 0 00 064640'                        MOVE    1,SAVRAR                ; get RAR address (for left 30 bits)
043272  620 01 0 00 000001                         TRZ     1,1                     ; clear bit 12 (for right 30 bits)
043273  260 17 0 00 043254*                        GO      LDRAR                   ; load address
043274  200 01 0 00 000003                         MOVE    1,3                     ; get right 30 bits
043275  336 00 0 00 064620'                        SKIPN   MVFLAG                  ; load?
043276  260 17 0 00 043266*                        GO      LDCRAM                  ; yes - load it
043277  336 00 0 00 064620'                        SKIPN   MVFLAG                  ; load?
043300  254 00 0 00 043322                         JRST    MLOAD1                  ; yes - next location
043301  260 17 0 00 040444*                        GO      RDCRAM                  ; yes - read CRAM
043302  202 01 0 00 037752                         MOVEM   1,CWORDR                ; save it
043303  316 02 0 00 037751                         CAMN    2,CWORDL                ; valid data?
043304  312 03 0 00 037752                         CAME    3,CWORDR
043305  334 00 0 00 000000                         SKIPA                           ; no - continue
043306  254 00 0 00 043322                         JRST    MLOAD1                  ; yes - next location
043307  350 01 0 00 064621'                        AOS     1,MVNUM                 ; get error count
043310  303 01 0 00 000002                         CAILE   1,2                     ; more than 3 errors?
043311  254 00 0 00 043322                         JRST    MLOAD1                  ; yes - next location
```

```
043312  200 00 0 00 037750                         MOVE    CADDR                   ; get address
```

```
043312  200 00 0 00 037750                  MOVE    CADDR                        ; get address
043313  202 00 0 01 043340                  MOVEM   MWBADR(1)                    ; save it
043314  202 02 0 01 043343                  MOVEM   2,MWBLCO(1)                  ; save left 30 bits (correct)
043315  202 03 0 01 043351                  MOVEM   3,MWBRCO(1)                  ; save right 30 bits (correct)
043316  200 02 0 00 037751                  MOVE    2,CWORDL                     ; get left 30 bits
043317  202 02 0 01 043346                  MOVEM   2,MWBLAC(1)                  ; save it (actual)
043320  200 02 0 00 037752                  MOVE    2,CWORDR                     ; get right 30 bits
043321  202 03 0 01 043354                  MOVEM   3,MWBRAC(1)                  ; save it (actual)

                                    ; Point to next microword

043322  350 00 0 00 000005          MLOAD1: AOS     5                            ; increment word count
043323  271 04 0 00 000002                  ADDI    4,2                          ; increment by 2
043324  254 00 0 00 043243                  JRST    MLOAD0                       ; loop till done

                                    ; Done - return count of words loaded

043325  337 00 0 00 064621'         MLOADX: SKIPG   MVNUM                        ; any verify errors?
043326  350 00 0 17 777773                  AOS     -5(P)                        ; no - set up RTN+2
043327  332 00 0 00 064622'                 SKIPE   MVREP                        ; report errors?
043330  260 17 0 00 043357                  GO      MVPNT                        ; yes - print errors
043331  202 05 0 00 064615'                 MOVEM   5,MLNUM#                     ; save count of words loaded
043332  262 17 0 00 000005                  RGET    (5,4,3,2,0)                  ; restore AC's
043337  263 17 0 00 000000                  RTN                                  ; return

043340                              MWBADR: BLOCK   3                            ; CRAM addresses
043343                              MWBLCO: BLOCK   3                            ; left correct
043346                              MWBLAC: BLOCK   3                            ; left actual
043351                              MWBRCO: BLOCK   3                            ; right correct
043354                              MWBRAC: BLOCK   3                            ; right actual
```

```
                                    ;***************************************************************************
```

```



                                                  ;#*******************************************************************
                                                  ;* MVPNT - Print verify error data
                                                  ;#*******************************************************************

043357  335 00 0 00 064621'   MVPNT:    SKIPGE  MVNUM                           ; any errors to report?
043360  263 17 0 00 000000              RTN                                     ; no - return
043361  261 17 0 00 000000              RPUT    (0,1,2,3,4,5,6)                 ; save some AC's
043370  350 01 0 00 064621'             AOS     1,MVNUM                         ; get correct error count 1..n
043371  200 00 0 00 000001              MOVE    1
043372  037 00 0 00 052573              TMSGC   <? >
043373  037 15 0 00 000000              PNTDEC                                  ; print it
043374  037 00 0 00 062205              TMSG    <. CRAM verify errors:>
043375  037 00 0 00 062212              TMSGC   <Addr    --- Correct ---          --- Actual ---       -- Discrepancy -->
043376  400 04 0 00 000000              SETZ    4,                              ; pointer to error data
043377  200 00 0 04 043340    MVPNT1:   MOVE    MWBADR(4)                       ; get address
043400  037 00 0 00 030242              PCRL
043401  037 04 0 00 000000              PNT4                                    ; print it
043402  037 00 0 00 053077              TMSG    < >
043403  200 02 0 04 043343              MOVE    2,MWBLCO(4)                     ; get left half
043404  200 03 0 04 043351              MOVE    3,MWBRCO(4)                     ; get right half
043405  260 17 0 00 043436              GO      MVPNTW                          ; print it
043406  037 00 0 00 053077              TMSG    < >
043407  200 02 0 04 043346              MOVE    2,MWBLAC(4)                     ; get left half
043410  200 03 0 04 043354              MOVE    3,MWBRAC(4)                     ; get right half
043411  260 17 0 00 043436              GO      MVPNTW                          ; print it
043412  037 00 0 00 053077              TMSG    < >
043413  200 02 0 04 043343              MOVE    2,MWBLCO(4)                     ; get left half
043414  430 02 0 04 043346              XOR     2,MWBLAC(4)                     ; xor actual data
043415  200 03 0 04 043351              MOVE    3,MWBRCO(4)                     ; get right half
043416  430 03 0 04 043354              XOR     3,MWBRAC(4)                     ; xor actual data
043417  260 17 0 00 043436              GO      MVPNTW                          ; print it
043420  377 00 0 00 000001              SOSG    1                               ; decrement error count - done?
043421  254 00 0 00 043426              JRST    MVPNTX                          ; yes - exit
043422  350 00 0 00 000004              AOS     4                               ; point to next word
043423  305 04 0 00 000003              CAIGE   4,3                             ; 3 printed already?
043424  254 00 0 00 043377              JRST    MVPNT1                          ; no - loop till done
043425  037 00 0 00 062230              TMSGC   <...>
043426  262 17 0 00 000006    MVPNTX:   RGET    (6,5,4,3,2,1,0)                 ; yes - restore AC's
043435  263 17 0 00 000000              RTN                                     ; return

043436  242 03 0 00 000006    MVPNTW:   LSH     3,^D6                           ; left justify the data
043437  246 02 0 00 000006              LSHC    2,^D6                           ; left justify both words
043440  200 05 0 00 062232              MOVE    5,[POINT 3,2]                   ; initial byte pointer
043441  201 06 0 00 000024              MOVEI   6,^D20                          ; number of bytes to print
043442  134 00 0 00 000005              ILDB    5                               ; get byte
043443  037 16 0 00 000003              PNTOCS                                  ; print it
043444  365 06 0 00 043442              SOJGE   6,.-2                           ; loop till done
043445  263 17 0 00 000000              RTN
```

```
                                                  ;*******************************************************************
```

```
                                             ;#*********************************************************************
                                             ;* MLIST - List microcode of a test
                                             ;#*********************************************************************

043446  261 17 0 00 000000          MLIST:   RPUT    (0,1,2,3,4)                 ; save some AC's
043453  037 00 0 00 061076                   TMSGC   <ADDR/ JUMP PAR OUT MGC SOR FUN DES CC RAM PA PB SK BUS CRY CTL  TM SP MK>
043454  200 04 0 00 000001                   MOVE    4,1                         ; get address of table in AC4
043455  200 02 0 04 000000          MLISTO:  MOVE    2,(4)                       ; get first word
043456  316 02 0 00 060261                   CAMN    2,[-1]                      ; end of list?
043457  254 00 0 00 043500                   JRST    MLISTX                      ; yes - exit
043460  200 03 0 04 000001                   MOVE    3,1(4)                      ; get second word

                                    ; Print this CRAM entry

043461  037 00 0 00 030242                   PCRL                                ; start a new line
043462  135 00 0 00 062200                   LDB     [POINT 12,2,11]             ; get cram load address
043463  001 04 0 00 000000                   PNTOCC  4,0                         ; print it
043464  037 00 0 00 061072                   TMSG    </ >
043465  200 01 0 00 000003                   MOVE    1,3                         ; get word 2 (it has 'bad parity' flag)
043466  135 02 0 00 062201                   LDB     2,[POINT 24,2,35]           ; get CRAM bits 0-23
043467  246 02 0 00 000006                   LSHC    2,6                         ; now get bits 0-30 in AC2 (left half)
043470  242 03 0 00 777772                   LSH     3,-6                        ; right justify AC3 (right half)
043471  622 03 0 00 000002                   TRZE    3,2                         ; bit 59 set?
043472  660 03 0 00 000001                   TRO     3,1                         ; yes - set rightmost bit
043473  260 17 0 00 043625                   GO      CALPAR                      ; calculate parity
043474  260 17 0 00 043507                   GO      PNTCRM                      ; print it

                                    ; Point to next microword / Exit if Altmode typed / Exit when done

043475  271 04 0 00 000002                   ADDI    4,2                         ; increment by 2
043476  336 00 0 00 064570'                  SKIPN   ALTF                        ; altmode typed?
043477  254 00 0 00 043455                   JRST    MLISTO                      ; no - loop till done
043500  037 00 0 00 030242          MLISTX:  PCRL
043501  262 17 0 00 000004                   RGET    (4,3,2,1,0)                 ; restore AC's
043506  263 17 0 00 000000                   RTN                                 ; return
```

```
                                             ;#*********************************************************************
```

```
                                                  ;#*****************************************************************
                                                  ;* PNTCRM - Print CRAM word
                                                  ;
                                                  ; Arguments:    AC2 - Left 30 bits
                                                  ;               AC3 - Right 30 bits
                                                  ;
                                                  ;#*****************************************************************

043507  261 17 0 00 000000       PNTCRM: RPUT     (0,1)                   ; save AC's
043511  135 00 0 00 062233               LDB      [POINT 12,2,17]         ; get MWJMPFLD
043512  001 04 0 00 000000               PNTOCC   4,0                     ; print it
043513  037 00 0 00 055371               TMSG     < >
043514  135 00 0 00 062234               LDB      [POINT 1,2,18]          ; get MWPAR
043515  037 01 0 00 000000               PNT1                             ; print it
043516  037 00 0 00 055371               TMSG     < >
043517  135 00 0 00 062235               LDB      [POINT 1,2,19]          ; get MWOUTPUTENA
043520  037 01 0 00 000000               PNT1                             ; print it
043521  135 00 0 00 062236               LDB      [POINT 10,2,29]         ; get MWMGCFLD
043522  001 04 0 00 000000               PNTOCC   4,0                     ; print it
043523  037 00 0 00 055371               TMSG     < >
043524  135 00 0 00 062237               LDB      [POINT 3,2,32]          ; get MWSORCEFLD
043525  037 01 0 00 000000               PNT1                             ; print it
043526  037 00 0 00 055371               TMSG     < >
043527  135 00 0 00 062240               LDB      [POINT 3,2,35]          ; get MWFUNCTFLD
043530  037 01 0 00 000000               PNT1                             ; print it
043531  037 00 0 00 055371               TMSG     < >
043532  135 00 0 00 062241               LDB      [POINT 3,3,8]           ; get MWDESTFLD
043533  037 01 0 00 000000               PNT1                             ; print it
043534  037 00 0 00 055371               TMSG     < >
043535  135 00 0 00 062242               LDB      [POINT 1,3,9]           ; get MWCCENA
043536  037 01 0 00 000000               PNT1                             ; print it
043537  037 00 0 00 000040               PSP
043540  135 00 0 00 062243               LDB      [POINT 1,3,10]          ; get MWRAMODE
043541  037 01 0 00 000000               PNT1                             ; print it
043542  037 00 0 00 000040               PSP
043543  135 00 0 00 062244               LDB      [POINT 4,3,14]          ; get MWPORTAFLD
043544  001 02 0 00 000000               PNTOCC   2,0                     ; print it
043545  037 00 0 00 000040               PSP
043546  135 00 0 00 062245               LDB      [POINT 4,3,18]          ; get MWPORTBFLD
043547  001 02 0 00 000000               PNTOCC   2,0                     ; print it
043550  037 00 0 00 000040               PSP
043551  135 00 0 00 062246               LDB      [POINT 5,3,23]          ; get MWSKIPFLD
043552  001 02 0 00 000000               PNTOCC   2,0                     ; print it
043553  037 00 0 00 055371               TMSG     < >
043554  135 00 0 00 062247               LDB      [POINT 3,3,26]          ; get MWBUSCTLFLD
043555  037 01 0 00 000000               PNT1                             ; print it
```

```
043556  037 00 0 00 055371               TMSG     < >
```

```
043556  037 00 0 00 055371          TMSG    < >
043557  135 00 0 00 062250          LDB     [POINT 1,3,27]          ; get MWCARRY
043560  037 01 0 00 000000          PNT1                            ; print it
043561  037 00 0 00 000040          PSP
043562  135 01 0 00 062251          LDB     1,[POINT 4,3,31]        ; get MWCTRLFLD
                                    XCT     [TMSG   <JZ  >          ; print in English
                                             TMSG   <CJS >
                                             TMSG   <JMAP>
                                             TMSG   <CJP >
                                             TMSG   <PUSH>
                                             TMSG   <JSRP>
                                             TMSG   <CJV >
                                             TMSG   <JRP >
                                             TMSG   <RFCT>
                                             TMSG   <RPCT>
                                             TMSG   <CRTN>
                                             TMSG   <CJPP>
                                             TMSG   <LDCT>
                                             TMSG   <LOOP>
                                             TMSG   <CONT>
043563  256 00 0 01 062272                   TMSG   <TWB >](1)
043564  037 00 0 00 055371          TMSG    < >
043565  135 00 0 00 062312          LDB     [POINT 1,3,32]          ; get MWTIMEFLD
043566  037 01 0 00 000000          PNT1                            ; print it
043567  037 00 0 00 000040          PSP
043570  135 00 0 00 062313          LDB     [POINT 2,3,34]          ; get MWSPARE00-01
043571  037 01 0 00 000000          PNT1                            ; print it
043572  037 00 0 00 000040          PSP
043573  135 00 0 00 062314          LDB     [POINT 1,3,35]          ; get MWMARKBIT
043574  037 01 0 00 000000          PNT1                            ; print it
043575  037 07 0 00 000003          TTALTM                          ; altmode typed?
043576  334 00 0 00 000000          SKIPA                           ; no - continue
                                    GO      [SETOM  ALTF            ; yes - set 'altmode typed' flag
                                             SETZM  MULFLG          ;   and clear 'examine next' flag
                                             RTN]
043577  260 17 0 00 062315          RGET    (1,0)                   ; restore AC's
043600  262 17 0 00 000001
043602  263 17 0 00 000000          RTN                             ; return
```

```
                                    ;#*********************************************************************
                                    ;* PNTWD - Print a CRAM word (xxxx xxxx xxxx xxxx xxxx)
                                    ;
                                    ; Argument:     AC2 - Left half
                                    ;               AC3 - Right half
                                    ;
                                    ;#*********************************************************************

                                    PNTWD:  RPUT    (0,1)                   ; save AC's
043603  261 17 0 00 000000                  LDB     [POINT 12,2,17]         ; get 1st 4 digits
043605  135 00 0 00 062233                  PNT4                            ; print them
043606  037 04 0 00 000000                  LDB     [POINT 12,2,29]         ; get 2nd 4 digits
043607  135 00 0 00 062320                  PNT4                            ; print them
043610  037 04 0 00 000000                  LDB     1,[POINT 6,2,35]        ; get next 6 bits
043611  135 01 0 00 062321                  LDB     [POINT 6,3,11]          ; get next 6 bits
043612  135 00 0 00 062322                  LSH     1,6                     ; position correctly
043613  242 01 0 00 000006                  IOR     1                       ; build 12 bits (3rd 4 digits)
043614  434 00 0 00 000001                  PNT4                            ; print them
043615  037 04 0 00 000000                  LDB     [POINT 12,3,23]         ; get 1st 4 digits
043616  135 00 0 00 062323                  PNT4                            ; print them
043617  037 04 0 00 000000                  LDB     [POINT 12,3,35]         ; get last 4 digits
043620  135 00 0 00 062324                  PNT4                            ; print them
043621  037 04 0 00 000000                  RGET    (1,0)                   ; restore AC's
043622  262 17 0 00 000001                  RTN                             ; return
043624  263 17 0 00 000000
```

```
                                    ;#*********************************************************************
                                    ;* CALPAR - Calculate parity for a CRAM location
                                    ;#*********************************************************************
                                    ;* CALPAR - Calculate parity for a CRAM location
                                    ;
                                    ; Arguments:    AC1 - Contains 2nd word of microword as  put  together
                                    ;                     by the microword macro MWORD.  This is only used
                                    ;                     to look at bit 35 which is a flag specifying bad
                                    ;                     parity.
                                    ;               AC2 - Bits 0-29 (Right justified)
                                    ;               AC3 - Bits 30-59 (Right justified)
                                    ;
                                    ; Function:     Calculate good parity and insert into Bit 12 of  the
                                    ;               microword which is Bit 18 in AC2. Then if 'force bad
                                    ;               parity' flag is set, complement this bit.
                                    ;
                                    ; Return:       +1 always.  AC's are unmodified, except bit 18 of AC2.
                                    ;#*********************************************************************

043625  261 17 0 00 000004          CALPAR: RPUT    (4,5,6,7)                       ; save AC's
043631  402 00 0 00 064572'                 SETZM   CALMAR#                         ; clear MARK bit location
043632  602 03 0 00 000001                  TRNE    3,1                             ; MARK bit set?
043633  476 00 0 00 064572'                 SETOM   CALMAR                          ; yes - note the fact
043634  620 03 0 00 000001                  TRZ     3,1                             ; ensure MARK bit is zero
043635  400 04 0 00 000000                  SETZ    4,                              ; clear count of number of 1's
043636  200 05 0 00 000002                  MOVE    5,2                             ; get left half
043637  210 06 0 00 000005                  MOVN    6,5                             ; calculate number of 1's
043640  632 05 0 00 000006                  TDZE    5,6                             ;  ...
043641  344 04 0 00 043637                  AOJA    4,.-2                           ;  ...
043642  200 05 0 00 000003                  MOVE    5,3                             ; get right half
043643  210 06 0 00 000005                  MOVN    6,5                             ; calculate number of 1's
043644  632 05 0 00 000006                  TDZE    5,6                             ;  ...
043645  344 04 0 00 043643                  AOJA    4,.-2                           ;  ...

                                    ; Now, the parity is even if bit 35 of AC4 is zero, and odd if bit 35 set

043646  606 04 0 00 000001                  TRNN    4,1                             ; bit 35 set?
043647  431 02 0 00 400000                  XORI    2,400000                        ; no - complement bit 18

                                    ; Also, handle force parity flag

043650  602 01 0 00 000001                  TRNE    1,1                             ; 'force bad parity' flag set?
043651  431 02 0 00 400000                  XORI    2,400000                        ; yes - complement bit 18

                                    ; Done - exit

043652  332 00 0 00 064572'                 SKIPE   CALMAR                          ; MARK bit set initially?
043653  660 03 0 00 000001                  TRO     3,1                             ; yes - set it now
043654  262 17 0 00 000007                  RGET    (7,6,5,4)                       ; restore AC's
043660  263 17 0 00 000000                  RTN                                     ; return
```

```
                                    ;#*********************************************************************
```

```
                                ;#*********************************************************************
                                ;* TLOAD - Load microcode/verify it for a test
                                ;
                                ; Arguments:    AC1 - Contains address where microcode starts
                                ;
                                ; Function:     Issue a port clear to the port to ensure it is stopped.
                                ;               Then call MLOAD to load the microcode, then call MVERFY
                                ;               to verify it.  Abort on any error.
                                ;
                                ; Returns:      +1 - Error occurred loading/verifying microcode
                                ;               +2 - Ok
                                ;#*********************************************************************

043661  332 00 0 00 064610'     TLOAD:  SKIPE   LDEBUG                  ; ucode load debug switch set?
043662  254 00 0 00 043701              JRST    TLOADX                  ; yes - don't load ucode
043663  316 16 0 00 047001              CAMN    MBCN,PORTNI             ; NI port?
                                        JRST    [CAMN   1,TSLOD1        ; yes - already loaded?
                                                 JRST   TLOADX          ; yes - don't load ucode
043664  254 00 0 00 062325                       JRST   .+1]
043665  316 16 0 00 047002              CAMN    MBCN,PORTCI             ; CI port?
                                        JRST    [CAMN   1,TSLOD2        ; yes - already loaded?
                                                 JRST   TLOADX          ; yes - don't load ucode
043666  254 00 0 00 062330                       JRST   .+1]
043667  261 17 0 00 000001              PUT     1                       ; save AC1
043670  260 17 0 00 036116*             GO      IPACLR                  ; do a 'port clear'
043671  260 17 0 00 043207              GO      MLOADN                  ; load it
043672  254 00 0 00 043703               JRST   TLOADE                  ; failed
043673  200 01 0 17 000000              MOVE    1,(P)                   ; get microcode address
043674  316 16 0 00 047001              CAMN    MBCN,PORTNI             ; NI port?
043675  202 01 0 00 064646'             MOVEM   1,TSLOD1                ; yes - save it
043676  316 16 0 00 047002              CAMN    MBCN,PORTCI             ; CI port?
043677  202 01 0 00 064647'             MOVEM   1,TSLOD2                ; yes - save it
043700  262 17 0 00 000001              GET     1                       ; restore AC1
043701  350 00 0 17 000000      TLOADX: AOS     (P)                     ; set up RTN +2
043702  263 17 0 00 000000              RTN                             ; exit

043703  262 17 0 00 000001      TLOADE: GET     1                       ; restore AC1
043704  474 15 0 00 000000              SETO    ERFLG,                  ; set error flag
043705  027 00 0 00 043713              SCOPER  TLERR                   ; print error message
043706  255 00 0 00 000000              JFCL                            ; don't allow test looping
043707  255 00 0 00 000000              JFCL                            ; ignore altmode
043710  332 00 0 00 064614'             SKIPE   MDEBUG                  ; error message debug switch set?
043711  350 00 0 17 000000              AOS     (P)                     ; yes - skip this error
043712  263 17 0 00 000000              RTN                             ; exit

043713  160000  062333          TLERR:  MSG!TXALL![ASCIZ /Error loading test microcode - test aborted./]
043714  270000  043357                  LAST!CALL!TXALL!MVPNT           ; print verify errors
```

```
                                ;#*********************************************************************
                                ;* CSRPNT - Print CSR register as 'CSR: xxxxxx,,xxxxxx' followed by
```

```

                                             ;#*********************************************************************
                                             ;* CSRPNT - Print CSR register as 'CSR: xxxxxx,,xxxxxx' followed by
                                             ;*        English translation.
                                             ;#*********************************************************************

043715  261 17 0 00 000000           CSRPNT: RPUT    (0,1,2,3)                ; save AC's
043721  037 00 0 00 062344                   TMSGC   <CSR: >                  ; print out the data
043722  200 00 0 00 000001                   MOVE    1
043723  037 13 0 00 000000                   PNTHW
043724  260 17 0 00 043111*                  GO      SWITT                    ; get switches
043725  603 00 0 00 000200                   TLNE    TXTINH                   ; TXTINH set?
043726  254 00 0 00 043732                   JRST    CSRPNX                   ; yes - exit
043727  201 02 0 00 000023                   MOVEI   2,^D19                   ; character count so far
043730  201 03 0 00 000023                   MOVEI   3,^D19                   ; continuation line column number
043731  260 17 0 00 043737                   GO      CSRENG                   ; go print in English
043732  262 17 0 00 000003           CSRPNX: RGET    (3,2,1,0)                ; restore AC's
043736  263 17 0 00 000000                   RTN                              ; return

                                             ;#*********************************************************************
                                             ;* CSRENG - Print CSR data in English
                                             ;
                                             ; AC1 - Contains data to print
                                             ; AC2 - Contains character count on line so far
                                             ; AC3 - Contains column to start continuation lines
                                             ;#*********************************************************************

043737  261 17 0 00 000000           CSRENG: RPUT    (0,1,4,5,6)              ; save AC's
043744  202 01 0 00 064574'                  MOVEM   1,CSRENS#                ; save CSR data
043745  200 04 0 00 000001                   MOVE    4,1                      ; get data into AC1
043746  400 05 0 00 000000                   SETZ    5,                       ; clear bit index
043747  404 04 0 00 062346                   AND     4,[777770,,777770]       ; clear port ID and PI bits
043750  434 04 0 00 062347                   IOR     4,[000004,,000004]       ; set bits so both are printed
043751  322 04 0 00 043770           CSREN0: JUMPE   4,CSRENX                 ; zero? yes - exit
043752  607 04 0 00 400000                   TLNN    4,400000                 ; no - bit set?
043753  254 00 0 00 043766                   JRST    CSREN2                   ; no - continue
043754  271 02 0 00 000007                   ADDI    2,7                      ; increment character count
043755  305 02 0 00 000110                   CAIGE   2,^D72                   ; over 72 characters?
043756  254 00 0 00 043765                   JRST    CSREN1                   ; no - go print
043757  037 00 0 00 030242                   PCRL                             ; yes - print CRLF, then
043760  200 01 0 00 000003                   MOVE    1,3                      ;   print spaces to start
043761  037 00 0 00 053077                   TMSG    < >                      ;   in proper column, and
043762  367 01 0 00 043761                   SOJG    1,.-1                    ;   adjust character count
043763  200 02 0 00 000003                   MOVE    2,3                      ;   properly
043764  271 02 0 00 000007                   ADDI    2,7                      ; increment character count
043765  256 00 0 05 043776           CSREN1: XCT     CSRENB(5)                ; yes - print it
043766  242 04 0 00 000001           CSREN2: LSH     4,1                      ; left shift one bit
043767  345 05 0 00 043751                   AOJGE   5,CSREN0                 ; increment and keep looping
043770  262 17 0 00 000006           CSRENX: RGET    (6,5,4,1,0)              ; restore AC's
043775  263 17 0 00 000000                   RTN
```

```

                                             ;#*********************************************************************
```

```
                                        ;#**********************************************************************
                                        ;* CSRENB - CSR bits
                                        ;#**********************************************************************

043776  037 00 0 00 062350              CSRENB: TMSG    <PortP >                ; 0
043777  037 00 0 00 062352                      TMSG    <CsrRqs >               ; 1
044000  037 00 0 00 062354                      TMSG    <CsrChn >               ; 2
044001  037 00 0 00 062356                      TMSG    <Dinit >                ; 3
044002  037 00 0 00 062360                      TMSG    <PI00Rq >               ; 4
044003  037 00 0 00 062362                      TMSG    <RqInt >                ; 5
044004  037 00 0 00 062364                      TMSG    <CramPE >               ; 6
044005  037 00 0 00 062366                      TMSG    <MBErr >                ; 7
044006  037 00 0 00 062370                      TMSG    <Unused >               ; 8
044007  037 00 0 00 062370                      TMSG    <Unused >               ; 9
044010  037 00 0 00 062370                      TMSG    <Unused >               ; 10
044011  037 00 0 00 062372                      TMSG    <Idle >                 ; 11
044012  037 00 0 00 062374                      TMSG    <Dcomp >                ; 12
044013  037 00 0 00 062376                      TMSG    <Ecomp >                ; 13
044014  037 00 0 00 062370                      TMSG    <Unused >               ; 14
                                                GO      [TMSG   <ID=>           ; 15
                                                         LDB    [POINT 3,CSRENS,17]
                                                         PNT1
                                                         RTN]
044015  260 17 0 00 062402
044016  255 00 0 00 000000                      JFCL                            ; 16
044017  255 00 0 00 000000                      JFCL                            ; 17
044020  037 00 0 00 062406                      TMSG    <PClr >                 ; 18
044021  037 00 0 00 062410                      TMSG    <TEbuf >                ; 19
044022  037 00 0 00 062412                      TMSG    <GenEPE >               ; 20
044023  037 00 0 00 062414                      TMSG    <SelLAR >               ; 21
044024  037 00 0 00 062416                      TMSG    <SinCyc >               ; 22
044025  037 00 0 00 062420                      TMSG    <Spare1 >               ; 23
044026  037 00 0 00 062422                      TMSG    <EbusPE >               ; 24
044027  037 00 0 00 062424                      TMSG    <FQErr >                ; 25
044030  037 00 0 00 062426                      TMSG    <MVErr >                ; 26
044031  037 00 0 00 062430                      TMSG    <CmdQAV >               ; 27
044032  037 00 0 00 062432                      TMSG    <ResQAV >               ; 28
044033  037 00 0 00 062434                      TMSG    <Spare2 >               ; 29
044034  037 00 0 00 062436                      TMSG    <Disabl >               ; 30
044035  037 00 0 00 062440                      TMSG    <Enable >               ; 31
044036  037 00 0 00 062442                      TMSG    <MPRun >                ; 32
                                                GO      [LDB    [POINT 3,CSRENS,35]
                                                         SKIPN
                                                         RTN
                                                         TMSG   <Pia=>          ; 33-35
                                                         PNT1
044037  260 17 0 00 062446                               RTN]
```

```
                                        ;#**********************************************************************
                                        ;* SAVCRM - Save specified CRAM locations in a buffer
```

```
                                              ;#*********************************************************************
                                              ;* SAVCRM - Save specified CRAM locations in a buffer
                                              ;#*********************************************************************

044040  261 17 0 00 000000      SAVCRM: RPUT    (0,1,2,3)               ; save AC's
044044  260 17 0 00 043670*             GO      IPACLR                  ; do a 'port clear'
044045  201 02 0 00 044107              MOVEI   2,SAVCRL                ; get address of storage area
044046  200 03 0 00 000001              MOVE    3,1                     ; get AOBJN word
044047  552 03 0 00 037750      SAVCRO: HRRZM   3,CADDR                 ; set up CRAM address
044050  260 17 0 00 040430              GO      DRCRAM                  ; read CRAM location
044051  120 00 0 00 037751              DMOVE   CWORDL                  ; get location
044052  124 00 0 02 000000              DMOVEM  (2)                     ; save it
044053  271 02 0 00 000002              ADDI    2,2                     ; point to next location
044054  253 03 0 00 044047              AOBJN   3,SAVCRO                ; loop till done
044055  262 17 0 00 000003              RGET    (3,2,1,0)               ; restore AC's
044061  263 17 0 00 000000              RTN                             ; return

                                              ;#*********************************************************************
                                              ;* RESCRM - Restore CRAM locations
                                              ;#*********************************************************************

044062  261 17 0 00 000000      RESCRM: RPUT    (0,1,2,3)               ; save AC's
044066  260 17 0 00 044044*             GO      IPACLR                  ; do a 'port clear'
044067  261 17 0 00 035764              PUT     PARFLG                  ; save parity flag
044070  402 00 0 00 035764              SETZM   PARFLG                  ; don't calculate parity
044071  201 02 0 00 044107              MOVEI   2,SAVCRL                ; get address of storage area
044072  200 03 0 00 000001              MOVE    3,1                     ; get AOBJN word
044073  552 03 0 00 037750      RESCRO: HRRZM   3,CADDR                 ; set up CRAM address
044074  120 00 0 02 000000              DMOVE   (2)                     ; save it
044075  124 00 0 00 037751              DMOVEM  CWORDL                  ; get location
044076  260 17 0 00 040376              GO      DWCRAM                  ; read CRAM location
044077  271 02 0 00 000002              ADDI    2,2                     ; point to next location
044100  253 03 0 00 044073              AOBJN   3,RESCRO                ; loop till done
044101  262 17 0 00 035764              GET     PARFLG                  ; restore parity flag
044102  262 17 0 00 000003              RGET    (3,2,1,0)               ; restore AC's
044106  263 17 0 00 000000              RTN                             ; return

                                ; CRAM storage area

044107                          SAVCRL: BLOCK   ^D50

```

```
                                        SUBTTL  SPEAR Reporting

                                              ;#*********************************************************************
                                        SUBTTL  SPEAR Reporting

                                ;#*********************************************************************
                                ;* SPREP1 - Print initial SPEAR report 'diagnostic started'
                                ;#*********************************************************************

044171  332 00 0 00 064654'     SPREP1: SKIPE   UDEBUG                  ; debug mode?
044172  263 17 0 00 000000              RTN                             ; yes - return
044173  332 00 0 00 030037              SKIPE   USER                    ; exec mode?
044174  332 00 0 00 064643'             SKIPE   SPEAR1                  ; already done?
044175  263 17 0 00 000000              RTN                             ; yes - return
044176  336 00 0 00 030516              SKIPN   MONTYP                  ; TOPS-10?
044177  263 17 0 00 000000              RTN                             ; yes - return

                                ; Check switches first

044200  261 17 0 00 000000              RPUT    (0,1)                   ; save AC's
044202  260 17 0 00 043724*             GO      SWITT                   ; get program switches
044203  602 00 0 00 004000              TRNE    DSPEAR                  ; SPEAR reporting disabled?
044204  254 00 0 00 044215              JRST    SPRE1X                  ; yes - exit

                                ; Get PPN and build entry / get time and insert it

044205  260 17 0 00 044404      SPCHK1: GO      GETPPN                  ; get user's PPN
044206  202 01 0 00 044232              MOVEM   1,SENT10+11+1           ; save in SPEAR entry block
044207  104 00 0 00 000227              GTAD                            ; get time and day
044210  202 01 0 00 044226              MOVEM   1,SENT10+5+1            ; save it

                                ; Make the entry

044211  201 01 0 00 044220              MOVEI   1,SENT10                ; get address of argument block
044212  201 02 0 00 000013              MOVEI   2,^D10+1                ; get length
044213  104 00 0 00 000527      S1:     SYERR                           ; write to SYSERR file

                                ; Exit

044214  476 00 0 00 064643'             SETOM   SPEAR1                  ; set 'SPEAR initial msg done' flag
044215  262 17 0 00 000001      SPRE1X: RGET    (1,0)                   ; restore AC's
044217  263 17 0 00 000000              RTN                             ; return

                                ; SPEAR entry block 10

044220  250000  000000          SENT10: 250000,,0                       ; event type
044221  000000  000000                  0                               ; zero
044222  000000  000000                  0                               ; zero
044223  000000  000000                  0                               ; zero
044224  000000  000000                  0                               ; (for Release 6.0 or later)
044225  000000  020002                  20002                           ; time of occurence block
044226  000000  000000                  0                               ; time
044227  000000  100004                  0,,100004                       ; 4 words, type 10 code
044230  44 46 60 64 41 00               SIXBIT  /DFPTA/                 ; diagnostic name
044231  000000  000004                  MCNVER,,DECVER                  ; diagnostic version
044232  000000  000000                  0                               ; user PPN
044233  000000  000000                  0
```

```
                                ;#*********************************************************************
                                ;* SPREP2 - Print error or successful completion SPEAR report
```

```

                                                   ;#*********************************************************************
                                                   ;* SPREP2 - Print error or successful completion SPEAR report
                                                   ;#*********************************************************************

044234  336 00 0 00 064654'      SPREP2: SKIPN   UDEBUG                  ; debug mode?
044235  336 00 0 00 030037               SKIPN   USER                    ; exec mode?
044236  263 17 0 00 000000               RTN                             ; yes - return
044237  336 00 0 00 030516               SKIPN   MONTYP                  ; TOPS-10?
044240  263 17 0 00 000000               RTN                             ; yes - return

                                 ; First check if anything should be reported now

044241  261 17 0 00 000000               PUT     0                       ; save AC0
044242  200 00 0 00 064644'              MOVE    SPEAR2                  ; get reporting flag
044243  322 00 0 00 044247               JUMPE   SPRE2A                  ; nothing yet? yes - continue
044244  316 00 0 00 060261               CAMN    [-1]                    ; already reported an error?
                                         JRST    [GET                    ; yes - exit
044245  254 00 0 00 055431                        RTN]
                                         JUMPE   ERFLG,[GET              ; error this time? no - exit
044246  322 15 0 00 055431                               RTN]

                                 ; Check switches

044247  261 17 0 00 000001       SPRE2A: RPUT    (1,2,3,4)               ; save AC's
044253  260 17 0 00 044202*              GO      SWITT                   ; get program switches
044254  602 00 0 00 004000               TRNE    DSPEAR                  ; SPEAR reporting disabled?
044255  254 00 0 00 044334               JRST    SPRE2X                  ; yes - exit

                                 ; Initialize entry

044256  402 00 0 00 044356       SPCHK2: SETZM   SENT11+^D11+1           ; clear test number
044257  402 00 0 00 044360               SETZM   SENT11+^D13+1           ;   ...
044260  201 02 0 00 000003               MOVEI   2,3                     ; set block length to 3 words
044261  137 02 0 00 062454               DPB     2,[POINT 9,SEN11L,35]   ; save length
044262  322 15 0 00 044320               JUMPE   ERFLG,SPRE2C            ; error? no - continue

                                 ; Insert test name

044263  200 01 0 00 047010               MOVE    1,TSTCLS                ; get test class
                                         MOVE    1,[SIXBIT /TSTE00/
                                                     SIXBIT /TSTS00/
                                                     SIXBIT /TSTA00/
                                                     SIXBIT /TSTM00/
044264  200 01 0 01 062455                           SIXBIT /TSTC00/](1)
044265  135 00 0 00 062462               LDB     [POINT 3,TSTNUM,32]     ; get digit 1
044266  271 00 0 00 000020               ADDI    20                      ; convert to SIXBIT
044267  137 00 0 00 062463               DPB     [POINT 6,1,29]          ; save it
044270  135 00 0 00 062464               LDB     [POINT 3,TSTNUM,35]     ; get digit 2
044271  271 00 0 00 000020               ADDI    20                      ; convert to SIXBIT
044272  137 00 0 00 062465               DPB     [POINT 6,1,35]          ; save it
044273  202 01 0 00 044356               MOVEM   1,SENT11+^D11+1         ; save test number
```

```

                                 ; Insert test description
```

```


                                                   ; Insert test description

        044274  200 02 0 00 047010                          MOVE     2,TSTCLS                 ; get test class
                                                            XCT      [MOVEI  2,TEBUS          ; get starting address in address
                                                                      MOVEI  2,TSEQ           ;   table
                                                                      MOVEI  2,TALU
                                                                      MOVEI  2,TMPROC
        044275  256 00 0 02 054210                                    MOVEI  2,TCBUS](2)
        044276  270 02 0 00 047011                          ADD      2,TSTNUM                 ; include the test number
        044277  370 00 0 00 000002                          SOS      2                        ; allow for tests 1..
        044300  202 02 0 00 047015                          MOVEM    2,TSTADD                 ; save the dispatch table address
        044301  260 17 0 00 042775                          GO       TSTGET
        044302  200 01 0 00 047007                          MOVE     1,TSTNAM                 ; get address of test description
        044303  474 02 0 00 000000                          SETO     2,                       ; initialize character count
        044304  200 03 0 00 062466                          MOVE     3,[POINT 7,0]            ; get initial byte pointer
        044305  540 03 0 00 000001                          HRR      3,1                      ; insert address
        044306  200 04 0 00 062467                          MOVE     4,[POINT 7,SENT11+^D13+1];get initial byte pointer
        044307  134 00 0 00 000003                 SPRE2B:  ILDB     3                        ; get a byte
        044310  136 00 0 00 000004                          IDPB     4                        ; save it
        044311  350 00 0 00 000002                          AOS      2                        ; increment byte count
        044312  326 00 0 00 044307                          JUMPN    SPRE2B                   ; loop till done

                                                   ; Insert block length ...

        044313  231 02 0 00 000005                          IDIVI    2,5                      ; calculate number of words
        044314  332 00 0 00 000003                          SKIPE    3                        ; remainder?
        044315  350 00 0 00 000002                          AOS      2                        ; yes - add in partial word
        044316  271 02 0 00 000003                          ADDI     2,3                      ; calculate subtotal
        044317  137 02 0 00 062454                          DPB      2,[POINT 9,SEN11L,35]    ; save length

                                                   ; Add time and day

        044320  104 00 0 00 000227                 SPRE2C:  GTAD                              ; get time and day
        044321  202 01 0 00 044350                          MOVEM    1,SENT11+5+1             ; save it

                                                   ; Get PPN and build entry / get time and insert it

        044322  260 17 0 00 044404                          GO       GETPPN                   ; get user's PPN
        044323  202 01 0 00 044354                          MOVEM    1,SEN11L-1               ; save in SPEAR entry block

                                                   ; Make the SPEAR entry

        044324  201 01 0 00 044342                          MOVEI    1,SENT11                 ; get address of argument block
        044325  135 02 0 00 062454                          LDB      2,[POINT 9,SEN11L,35]    ; get length
        044326  271 02 0 00 000013                          ADDI     2,^D10+1                 ; calculate total block length
        044327  104 00 0 00 000527                 S2:      SYERR                             ; write to SYSERR file
```

```

                                                   ; Exit
```

epancy -->

```

                                        ; Exit

044330  200 00 0 00 060261                       MOVE     [-1]                     ; get flag for 'Error message'
044331  336 00 0 00 000015                       SKIPN    ERFLG                    ; error flag set?
044332  200 00 0 00 061340                       MOVE     [-2]                     ; no - get flag for 'Final message'
044333  202 00 0 00 064644'                      MOVEM    SPEAR2                   ; set 'SPEAR err/fin msg done' flag
044334  262 17 0 00 000004              SPRE2X:  RGET     (4,3,2,1,0)              ; restore AC's
044341  263 17 0 00 000000                       RTN                               ; return

                                        ; SPEAR entry block 11

044342  250000  000000                  SENT11:  250000,,0                         ; event type
044343  000000  000000                           0                                 ; zero
044344  000000  000000                           0                                 ; zero
044345  000000  000000                           0                                 ; zero
044346  000000  000000                           0                                 ; (for Release 6.0 or later)
044347  000000  020002                           20002                             ; time of occurence block
044350  000000  000000                           0                                 ; time

044351  000000  100004                           0,,100004                         ; 4 words, type 10 code
044352  44 46 60 64 41 00                        SIXBIT /DFPTA/                    ; diagnostic name
044353  000000  000004                           MCNVER,,DECVER                    ; diagnostic version
044354  000000  000000                           0                                 ; user PPN

044355  000000  110000                  SEN11L:  0,,110000                         ; type 11 code
044356  000000  000000                           0                                 ; failing test name
044357  000000  000003                           3                                 ; failing test description offset
044360                                           BLOCK    ^D20                     ; ASCII test description


                                        ;#*********************************************************************
                                        ;* GETPPN - Find out PPN number
                                        ;#*********************************************************************

044404  261 17 0 00 000002              GETPPN:  RPUT     (2,3)                    ; save AC's
044406  474 01 0 00 000000                       SETO     1,                       ; indicate current job
044407  200 02 0 00 062470                       MOVE     2,[-1,,1]                ; place 1 word into AC1
044410  201 03 0 00 000003                       MOVEI    3,3                      ; point to connected directory number
044411  104 00 0 00 000507                       GETJI                             ; get job information
044412  400 01 0 00 000000                       SETZ     1,                       ; error - just clear AC1
044413  262 17 0 00 000003                       RGET     (3,2)                    ; restore AC's
044415  263 17 0 00 000000                       RTN                               ; return

```

```
                                                 SUBTTL  Test Execute Routines

```

```
                SUBTTL  Test Execute Routines

;#***********************************************************************
;* SEXEC - Execute entries in a 'Single Step Table'
;
; This routine is used to simplify the coding of micro-diagnostic tests
; which single step the microsequencer and which normally  pass or fail
; depending on the CRAM address latched into the Latch Address Register
; at various points in the test.
;
; This routine is given a pointer to the 'Single Step Table'.  It then
; executes one entry and returns +1,+2,+3 or +4 depending  on the type
; of entry and the result.
;
; Argument:     AC6 - Address of 'Single Step Table'
;
; Return:       +1 - End of table was seen.
;               +2 - A routine was called, and this entry has completed.
;               +3 - The single steps are done but final address incorrect
;               +4 - The single steps are done and final address correct
;
;               NSSTEP - Contains number of single steps done
;               SSADDR - Initial starting address
;               SEADDR - Correct final address
;               SAADDR - Actual final address
;
;               AC6 - Updated to point to the next table location
;
; Single Step Table Format:
;
;               Entry
;               Entry
;               ...
;               SSLAST          ; last entry (zero)
;
; 'Entry' is of the form:
;
; Bits 0-2 - Type of entry - 0 - Single Step (start addr given)(SSSTRT)
;                            1 - Single Step (continue)         (SSCONT)
;                            2 - Special Call                   (SSCALL)
;                            3 - Special Call, pass/fail exit   (SSCHK)
;                            4 - Select new table address       (SSJRST)
```

```

; Bits 3-35 - Defined per entry type.
```

```
                                  ; Bits 3-35 - Defined per entry type.
                                  ;
                                  ;    Entry type 0 (SSLAST) - All zero - indicates end of table
                                  ;
                                  ;    Entry type 1 (SSSTRT) - Bits 3-11 - Single step address to start at
                                  ;                                12-23 - Number of single steps to do
                                  ;                                24-35 - Correct end address
                                  ;
                                  ;    Entry type 2 (SSCONT) - Bits 3-11 - Unused
                                  ;                                12-23 - Number of single steps to do
                                  ;                                24-35 - Correct end address
                                  ;
                                  ;    Entry type 3 (SSCALL) - Bits 3-35 - Address of routine to call
                                  ;
                                  ;    Entry type 4 (SSCHK) -  Bits 3-35 - Address of routine to call (return
                                  ;                                        +1 if done with test, +2 otherwise)
                                  ;
                                  ;    Entry type 5 (SSJRST) - Bits 3-35 - Table address to continue
                                  ;********************************************************************

044416  261 17 0 00 000000        SEXEC:  RPUT    (0,1,2)                 ; save AC's

                                  ; Check if end of table

044421  200 02 0 06 000000                MOVE    2,(6)                   ; get entry
044422  350 00 0 00 000006                AOS     6                       ; point to next entry
044423  322 02 0 00 044427                JUMPE   2,SEXEX1                ; end of table - yes - exit

                                  ; Determine type of entry and dispatch

044424  135 01 0 00 062471                LDB     1,[POINT 3,2,2]         ; get entry
                                          XCT     [JRST   SEXEX1          ; dispatch on it
                                                   JRST   SEXSSS
                                                   JRST   SEXSSC
                                                   JRST   SEXCAL
                                                   JRST   SEXCHK
                                                   JRST   SEXJRS
                                                   JRST   SEXERR
                                                   JRST   SEXERR
                                                   JRST   SEXERR
044425  256 00 0 01 062472                         JRST   SEXERR](1)

                                  ; Error in Single Step Table

044426  037 00 0 00 062504        SEXER   [illegible]CD  <Single Step Dispatch Table Error - Entry out of range>
044427  332 00 0 00 064641'       SEXEX1  SKIPE   SDEBUG                  ; EXEC debug mode?
044430  037 00 0 00 030242                PCRL                            ; yes - a final CRLF
044431  262 17 0 00 000002                RGET    (2,1,0)                 ; restore AC's
044434  263 17 0 00 000000                RTN                             ; return
```

```
                                  ; Execute single step
```

```

                                ; Execute single step

044435  135 00 0 00 062520      SEXSSS: LDB     [POINT 12,2,23]         ; get start address
044436  202 00 0 00 042750*             MOVEM   SNEXT                   ; save it
044437  200 00 0 00 044436*     SEXSSC: MOVE    SNEXT                   ; get next address
044440  202 00 0 00 044551              MOVEM   SSADDR                  ; save as start address
044441  135 00 0 00 062521              LDB     [POINT 9,2,11]          ; get # of single steps to do
044442  202 00 0 00 044550              MOVEM   NSSTEP                  ; save it
044443  135 00 0 00 062522              LDB     [POINT 12,2,35]         ; get correct end address
044444  202 00 0 00 044552              MOVEM   SEADDR                  ; save it
044445  350 00 0 00 047012              AOS     TSTSUB                  ; increment test segment

                                ; Set up start data

044446  336 01 0 00 042751*             SKIPN   1,SDATA                 ; any data given?
044447  201 01 0 00 000014              MOVEI   1,MPRUN!4               ; no - set initial data
044450  332 00 0 00 030037              SKIPE   USER                    ; user mode?
044451  620 01 0 00 000007              TRZ     1,7                     ; yes - clear PI level bits
044452  660 01 0 00 060010              TRO     1,SELLAR!MPRUN!SINCYC   ; ensure SELLAR,MPRUN,SINCYC are set
044453  202 01 0 00 044446*             MOVEM   1,SDATA                 ; save start data
044454  260 17 0 00 000000*             GO      SETLAR                  ; set up to write to LAR
044455  255 00 0 00 000000               JFCL                           ; error

                                ; Single step NSSTEP times

044456  200 02 0 00 044550              MOVE    2,NSSTEP                ; get number of times

                                ; First write the LAR

044457  200 01 0 00 044437*     SEXSS0: MOVE    1,SNEXT                 ; get start address
044460  242 01 0 00 000001              LSH     1,1                     ; position properly
044461  260 17 0 00 043273*             GO      LDRAR                   ; load the register

                                ; Then do the single step

044462  200 01 0 00 044453*             MOVE    1,SDATA                 ; get start data
044463  260 17 0 00 042337*             GO      LDCSR                   ; write to CSR

                                ; Then read the LAR

044464  260 17 0 00 042340*             GO      RDLAR                   ; read LAR
044465  242 01 0 00 777777              LSH     1,-1                    ; position properly
044466  202 01 0 00 044457*             MOVEM   1,SNEXT                 ; set up new next address
044467  367 02 0 00 044457              SOJG    2,SEXSS0                ; loop till done
044470  202 01 0 00 044553              MOVEM   1,SAADDR                ; save actual end address
```

```
                                        ; Debug output

 044471  332 00 0 00 064641'                  SKIPE    SDEBUG                     ; EXEC debug mode?
                                              GO       [TMSGC  <SSTEP:  #S >      ; yes - print debug
                                                        MOVE   NSSTEP              ;   output and then
                                                        PNTDEC                     ;   continue
                                                        TMSG   <, SADR >
                                                        MOVE   SSADDR
                                                        PNTOCS
                                                        TMSG   <, EADR >
                                                        MOVE   SEADDR
                                                        PNTOCS
                                                        TMSG   <, AADR >
                                                        MOVE   SAADDR
                                                        PNTOCS
 044472  260 17 0 00 062534                              RTN]

                                        ; Done - compare end address to expected

 044473  200 01 0 00 044553                   MOVE     1,SAADDR                   ; save actual end address
 044474  316 01 0 00 044552                   CAMN     1,SEADDR                   ; correct result?
 044475  350 00 0 17 777775                   AOS      -3(P)                      ; yes - increment return
 044476  350 00 0 17 777775                   AOS      -3(P)                      ; set up return
 044477  350 00 0 17 777775                   AOS      -3(P)                      ; set up return
 044500  262 17 0 00 000002                   RGET     (2,1,0)                    ; restore AC's
 044503  263 17 0 00 000000                   RTN                                 ; return +3/+4

                                        ; Execute a JRST and exit

 044504  550 02 0 00 000002             SEXJRS: HRRZ   2,2                        ; only want the address
 044505  332 00 0 00 064641'                  SKIPE    SDEBUG                     ; EXEC debug mode?
                                              GO       [TMSGC  <SJRST Adr  >      ; yes - print debug
                                                        MOVE   2                   ;   data and continue
                                                        PNTOCS
 044506  260 17 0 00 062554                              RTN]
 044507  200 06 0 00 000002                   MOVE     6,2                        ; get new table address
 044510  262 17 0 00 000002                   RGET     (2,1,0)                    ; restore AC's
 044513  350 00 0 17 000000                   AOS      (P)                        ; set up return
 044514  263 17 0 00 000000                   RTN                                 ; return +2
```

```
                                        ; Execute a CALL and exit
```

```
                                ; Execute a CALL and exit

044515  550 02 0 00 000002,     SEXCAL: HRRZ    2,2                     ; only want the address
044516  332 00 0 00 064641'             SKIPE   SDEBUG                  ; EXEC debug mode?
                                        GO      [TMSGC  <SCALL Adr  >   ; yes - print debug
                                                 MOVE    2              ;   data and continue
                                                 PNTOCS
                                                 RTN]
044517  260 17 0 00 062563              RPUT    (3,4)                   ; save some AC's
044520  261 17 0 00 000003              GO      @2                      ; call the routine
044522  260 17 1 00 000002              RGET    (4,3,2,1,0)             ; restore AC's
044523  262 17 0 00 000004              AOS     (P)                     ; set up return
044530  350 00 0 17 000000              RTN                             ; return +2
044531  263 17 0 00 000000

                                ; Execute a CALL and exit (if the return is +1 then exit test)

044532  550 02 0 00 000002,     SEXCHK: HRRZ    2,2                     ; only want the address
044533  332 00 0 00 064641'             SKIPE   SDEBUG                  ; EXEC debug mode?
                                        GO      [TMSGC  <SCHK Adr  >    ; yes - print debug
                                                 MOVE    2              ;   data and continue
                                                 PNTOCS
                                                 RTN]
044534  260 17 0 00 062572              RPUT    (3,4)                   ; save some AC's
044535  261 17 0 00 000003              GO      @2                      ; call the routine
044537  260 17 1 00 000002              SKIPA                           ; exit test
044540  334 00 0 00 000000              AOS     -5(P)                   ; ok - set up return
044541  350 00 0 17 777773              RGET    (4,3,2,1,0)             ; restore AC's
044542  262 17 0 00 000004              RTN                             ; return +2 or +1
044547  263 17 0 00 000000

                                ; Miscellaneous

044550  000000  000000          NSSTEP: 0                               ; number of single steps to do
044551  000000  000000          SSADDR: 0                               ; initial starting address
044552  000000  000000          SEADDR: 0                               ; correct final address
044553  000000  000000          SAADDR: 0                               ; actual final address
```

```
                                ;#***********************************************************************
                                ;* SSPNT - Print out single step data that resulted in an error
```

```
                                       ;#*********************************************************************
                                       ;* SSPNT - Print out single step data that resulted in an error
                                       ;#*********************************************************************

044554  261 17 0 00 000000             SSPNT:  RPUT    (0,1)
044556  200 01 0 00 000000*                    MOVE    1,SCOSW                 ; get switches
044557  603 01 0 00 000200                     TLNE    1,TXTINH                ; inhibit text?
044560  254 00 0 00 044577                     JRST    SSPN1                   ; yes - print alternate format

                                       ; Print single step data

044561  037 00 0 00 062576                     TMSGC   <Start Addr:  >
044562  200 00 0 00 044551                     MOVE    SSADDR
044563  037 04 0 00 000000                     PNT4
044564  037 00 0 00 062602                     TMSG    <, # Steps:  >
044565  200 00 0 00 044550                     MOVE    NSSTEP
044566  037 15 0 00 000000                     PNTDEC
044567  037 00 0 00 057306                     TMSG    <.>
044570  037 00 0 00 062605                     TMSGC   <End Addr - Correct:  >
044571  200 00 0 00 044552                     MOVE    SEADDR
044572  037 04 0 00 000000                     PNT4
044573  037 00 0 00 062612                     TMSGC   <              Actual:  >
044574  200 00 0 00 044553                     MOVE    SAADDR
044575  037 04 0 00 000000                     PNT4
044576  254 00 0 00 044614                     JRST    SSPNX

                                       ; Print single step data

044577  037 00 0 00 062617             SSPN1:  TMSGC   <Start  >
044600  200 00 0 00 044551                     MOVE    SSADDR
044601  037 04 0 00 000000                     PNT4
044602  037 00 0 00 062621                     TMSG    <, # Steps  >
044603  200 00 0 00 044550                     MOVE    NSSTEP
044604  037 15 0 00 000000                     PNTDEC
044605  037 00 0 00 057306                     TMSG    <.>
044606  037 00 0 00 062624                     TMSGC   <End (C):  >
044607  200 00 0 00 044552                     MOVE    SEADDR
044610  037 04 0 00 000000                     PNT4
044611  037 00 0 00 062627                     TMSGC   <    (A):  >
044612  200 00 0 00 044553                     MOVE    SAADDR
044613  037 04 0 00 000000                     PNT4

                                       ; Done - exit

044614  262 17 0 00 000001             SSPNX:  RGET    (1,0)                   ; restore AC's
044616  263 17 0 00 000000                     RTN
```

```
                                       ;#*********************************************************************
                                       ;* AEXEC - ALU type test execute routine
```

```
                                  ;#*********************************************************************
                                  ;* AEXEC - ALU type test execute routine
                                  ;
                                  ; This routine very simply starts up the port and waits for it to
                                  ; stop, then reads the LAR to find out where it  stopped.  If  it
                                  ; does not stop (times out), this routine forces it to  stop  and
                                  ; records the stopping address.
                                  ;
                                  ; Arguments:    AC1 - Amount of time in milliseconds before timeout
                                  ;               AC2 - Correct error address
                                  ;               AC3 - Correct final address
                                  ;               AC4 - Start address
                                  ;
                                  ; Results:      Rtn +1 - Error occurred
                                  ;               Rtn +2 - No error found
                                  ;
                                  ; Updated:      ALSRT - start address
                                  ;               ALCSR - final CSR data
                                  ;               ALERR - error final address
                                  ;               ALCOR - correct final address
                                  ;               ALACT - actual final address
                                  ;               ALEBF - contents of EBUF at end
                                  ;               ALTIM - amount of time to wait before timeout
                                  ;               ALFLS - flag - start up (0 ok, -1 failed)
                                  ;               ALFLE - flag - error address (0 ok, -1 incorrect)
                                  ;               ALFLC - flag - stopped with CRAM PE (0 yes, -1 no)
                                  ;               ALFLT - flag - timeout occurred (0 no, -1 yes)
                                  ;#*********************************************************************

044617  201 17 0 00 000000        AEXEC:  RPUT    (0,1,2)                 ; save some AC's

                                  ; Initialize everything

044622  202 01 0 00 044717                MOVEM   1,ALTIM                 ; save timeout time
044623  202 02 0 00 044723                MOVEM   2,ALERR                 ; save error final address
044624  202 03 0 00 044724                MOVEM   3,ALCOR                 ; correct final address
044625  202 04 0 00 044720                MOVEM   4,ALSRT                 ; save start address
044626  202 04 0 00 044466*               MOVEM   4,SNEXT                 ; set up start address
044627  402 00 0 00 044721                SETZM   ALCSR                   ; clear final CSR data
044630  402 00 0 00 044725                SETZM   ALACT                   ; clear actual final address
044631  402 00 0 00 044726                SETZM   ALFLS                   ; clear flag 'start up ok'
044632  402 00 0 00 044727                SETZM   ALFLE                   ; clear flag 'error address ok'
044633  402 00 0 00 044730                SETZM   ALFLC                   ; clear flag 'stopped with CRAM PE'
044634  402 00 0 00 044731                SETZM   ALFLT                   ; clear flag 'timeout occurred'
```

```

                                  ; Start up the port
```

```
                                            ; Start up the port

044635  260 17 0 00 044066*                         GO      IPACLR                  ; do a 'port clear'
044636  260 17 0 00 044454*                         GO      SETLAR                  ; set up to write the LAR
                                                     JRST   [SETOM  ALFLS           ; failed - set flag and
044637  254 00 0 00 062632                                   JRST   AEXECX]         ;   exit
044640  200 01 0 00 044626*                         MOVE    1,SNEXT                 ; get start address
044641  242 01 0 00 000001                          LSH     1,1                     ; position properly
044642  260 17 0 00 044461*                         GO      LDRAR                   ; write to RAR
044643  201 01 0 00 000010                          MOVEI   1,MPRUN                 ; get MPRUN bit
044644  260 17 0 00 044463*                         GO      LDCSR                   ; start up the port

                                            ; Wait the amount of time specified by the timeout period

044645  200 00 0 00 044717                          MOVE    ALTIM                   ; get amount of time
044646  260 17 0 00 043121                          GO      ODELAY                  ; wait

                                            ; Check final conditions

044647  260 17 0 00 043176*                         GO      RDCSR                   ; read CSR
                                                     J      [SETOM  ALFLS           ; failed - set flag and
044650  254 00 0 00 062632                                   JRST   AEXECX]         ;   exit
044651  202 01 0 00 044721                          MOVEM   1,ALCSR                 ; save final CSR data
044652  607 01 0 00 004000                          TLNN    1,(CRAMPE)              ; get a CRAM PE?
044653  476 00 0 00 044730                          SETOM   ALFLC                   ; no - flag it
044654  606 01 0 00 000010                          TRNN    1,MPRUN                 ; MPRUN still set?
044655  254 00 0 00 044667                          JRST    AEXECO                  ; no - continue
044656  476 00 0 00 044731                          SETOM   ALFLT                   ; yes - flag timeout occurred
044657  400 01 0 00 000000                          SETZ    1,                      ; write 0's to CSR
044660  260 17 0 00 044644*                         GO      LDCSR                   ; this should stop it
044661  260 17 0 00 044647*                         GO      RDCSR                   ; read CSR
044662  334 00 0 00 000000                           SKIPA                          ; error
044663  602 01 0 00 000010                          TRNE    1,MPRUN                 ; MPRUN still set?
044664  260 17 0 00 044635*                         GO      IPACLR                  ; yes - do a port clear
044665  260 17 0 00 044661*                         GO      RDCSR                   ; read it again
044666  255 00 0 00 000000                           JFCL                           ; error

                                            ; Read EBUF

044667  201 01 0 00 200000                  AEXECO: MOVEI   1,TSTEBF                ; set bit to read EBUF
044670  260 17 0 00 044660*                         GO      LDCSR                   ; write it
044671  260 17 0 00 041041*                         GO      RDEBUF                  ; read EBUF
044672  202 01 0 00 044722                          MOVEM   1,ALEBF                 ; save data
```

```

                                            ; Read LAR ...
```

```

                                        ; Read LAR ...

 044673  260 17 0 00 044636*                    GO      SETLAR                          ; set up to read LAR
 044674  255 00 0 00 000000                      JFCL                                   ; ignore error
 044675  260 17 0 00 044464*                    GO      RDLAR                           ; read it
 044676  242 01 0 00 777777                     LSH     1,-1                            ; position properly
 044677  202 01 0 00 044725                     MOVEM   1,ALACT                         ; save final address
 044700  312 01 0 00 044724                     CAME    1,ALCOR                         ; correct?
 044701  476 00 0 00 044727                     SETOM   ALFLE                           ; no - flag it

                                        ; Exit

 044702  262 17 0 00 000002             AEXECX: RGET    (2,1,0)                         ; restore AC's
 044705  332 00 0 00 044727                     SKIPE   ALFLE                           ; error flag set?
 044706  263 17 0 00 000000                     RTN                                     ; yes - return +1
 044707  332 00 0 00 044726                     SKIPE   ALFLS                           ; error flag set?
 044710  263 17 0 00 000000                     RTN                                     ; yes - return +1
 044711  332 00 0 00 044730                     SKIPE   ALFLC                           ; error flag set?
 044712  263 17 0 00 000000                     RTN                                     ; yes - return +1
 044713  332 00 0 00 044731                     SKIPE   ALFLT                           ; error flag set?
 044714  263 17 0 00 000000                     RTN                                     ; yes - return +1
 044715  350 00 0 17 000000                     AOS     (P)                             ; no - set up RTN+2
 044716  263 17 0 00 000000                     RTN                                     ; return +2

                                        ; Test variables

 044717  000000  000000                 ALTIM:  0                                       ; timeout time
 044720  000000  000000                 ALSRT:  0                                       ; start address
 044721  000000  000000                 ALCSR:  0                                       ; final CSR data
 044722  000000  000000                 ALEBF:  0                                       ; final EBUF data
 044723  000000  000000                 ALERR:  0                                       ; error final address
 044724  000000  000000                 ALCOR:  0                                       ; correct final address
 044725  000000  000000                 ALACT:  0                                       ; actual final address
 044726  000000  000000                 ALFLS:  0                                       ; flag (0 - started ok
                                                                                        ;       -1 - failed to start)
 044727  000000  000000                 ALFLE:  0                                       ; flag (0 - error address ok
                                                                                        ;       -1 - incorrect err addr)
 044730  000000  000000                 ALFLC:  0                                       ; flag (0 - stopped with CRAM PE
                                                                                        ;       -1 - did not have CRAM PE)
 044731  000000  000000                 ALFLT:  0                                       ; flag (0 - no timeout
                                                                                        ;       -1 - timeout occurred)
```

```

                                        ;#*********************************************************************
                                        ;* AAPNT - Print results message
```

```
                                                ;#*********************************************************************
                                                ;* AAPNT - Print results message
                                                ;#*********************************************************************

044732  261 17 0 00 000000                      AAPNT:  RPUT    (0,1)                   ; save AC's
044734  037 00 0 00 062634                              TMSGC   <Start addr:  >
044735  200 00 0 00 044720                              MOVE    ALSRT                   ; get start address
044736  037 04 0 00 000000                              PNT4                            ; print it
044737  200 01 0 00 044556*                             MOVE    1,SCOSW                 ; get switches
044740  607 01 0 00 000200                              TLNN    1,TXTINH                ; full printout?
                                                        GO      [TMSGC  <Final CSR data: >
                                                                 MOVE   ALCSR           ; yes - get CSR data
                                                                 PNTHW                  ; print it
                                                                 TMSGC  <End Addr - Correct:  >
                                                                 MOVE   ALCOR
                                                                 PNT4
                                                                 TMSGC  <               Actual:  >
                                                                 RTN]
044741  260 17 0 00 062644                              TLNE    1,TXTINH                ; full printout?
044742  603 01 0 00 000200                              GO      [TMSGC  <End Addr (C):  >
                                                                 MOVE   ALCOR
                                                                 PNT4
                                                                 TMSGC  <          (A):  >
                                                                 RTN]
044743  260 17 0 00 062664                              MOVE    ALACT                   ; get end address
044744  200 00 0 00 044725                              PNT4                            ; print it
044745  037 04 0 00 000000

                                                ; Message if never started up in the first place

044746  336 00 0 00 044726                              SKIPN   ALFLS                   ; never got it started?
044747  254 00 0 00 044752                              JRST    AAPNT0                  ; no - continue
044750  037 00 0 00 062671                              TMSGC   <Couldn't start the port>
044751  254 00 0 00 044765                              JRST    AAPNTX                  ; exit

                                                ; Message if stopped in error at the wrong place

044752  336 00 0 00 044727                      AAPNT0: SKIPN   ALFLE                   ; error addr correct?
044753  254 00 0 00 044763                              JRST    AAPNT1                  ; yes - continue
044754  603 01 0 00 000200                              TLNE    1,TXTINH                ; full printout?
044755  254 00 0 00 044765                              JRST    AAPNTX                  ; no - exit
044756  037 00 0 00 062677                              TMSGC   <(Should have stopped at >
044757  200 00 0 00 044724                              MOVE    ALCOR                   ; get correct address
044760  037 04 0 00 000000                              PNT4                            ; print it
044761  037 00 0 00 062705                              TMSG    < - test did not function properly)>
044762  254 00 0 00 044765                              JRST    AAPNTX                  ; exit
```

```
                                                ; Message if CRAM PE did not set also
```

```

                                         ; Message if CRAM PE did not set also

044763  332 00 0 00 044730               AAPNT1: SKIPE   ALFLC                   ; CRAM PE set?
044764  037 00 0 00 062714                       TMSGC   <CRAM PE should have set upon error>

                                         ; Exit

044765  262 17 0 00 000001               AAPNTX: RGET    (1,0)                   ; restore AC's
044767  263 17 0 00 000000                       RTN                             ; exit
```

```

                                         ;#**********************************************************************
                                         ;* BEXEC - Execute entries in a 'Single Step Table', verifying EBUF data
```

```
;#**********************************************************************
;* BEXEC - Execute entries in a 'Single Step Table', verifying EBUF data
;
; This routine is used to simplify the coding of micro-diagnostic tests
; which single step the microsequencer and which normally  pass or fail
; depending  on  the result stored in the EBUF at various points in the
; test.
;
; This routine is given a pointer to the 'Single Step Table'.  It then
; executes one entry and returns +1,+2,+3 or +4 depending  on the type
; of entry and the result.
;
; Argument:      AC6 - Address of 'Single Step Table'
;
; Return:       +1 - End of table was seen.
;               +2 - A routine was called, and this entry has completed.
;               +3 - The single steps are done but EBUF data incorrect
;               +4 - The single steps are done and EBUF data correct
;
;                NSSTEP - Contains number of single steps done
;                SSADDR - Initial starting address
;                SEADDR - Correct final address
;                SAADDR - Actual final address
;                CEBUF - Correct final EBUF data
;                AEBUF - Actual final EBUF data
;                BFLAG - flag - error address (0 ok, -1 incorrect)
;
;                AC6 - Updated to point to the next table location
;
; Single Step Table Format:
;
;                Entry
;                Entry
;                ...
;                SSLAST          ; last entry (zero)
;
; 'Entry' is of the form:
;
; Bits 0-2 - Type of entry - 0 - Single Step (start addr given)(SSSTRT)
;                            1 - Single Step (continue)        (SSCONT)
;                            2 - Special Call                  (SSCALL)
;                            3 - Special Call, pass/fail exit  (SSCHK)
;                            4 - Select new table address      (SSJRST)
;
```

```
; Bits 3-35 - Defined per entry type.
;
```

```
                                         ; Bits 3-35 - Defined per entry type.
                                         ;
                                         ;    Entry type 0 (SSLAST) - All zero - indicates end of table
                                         ;
                                         ;    Entry type 1 (SSSTRT) - Bits 3-11 - Number of single steps to do
                                         ;                                12-23 - Single step address to start at
                                         ;                                24-35 - Correct end address
                                         ;
                                         ;    Entry type 2 (SSCONT) - Bits 3-11 - Number of single steps to do
                                         ;                                12-23 - Unused
                                         ;                                24-35 - Correct end address
                                         ;
                                         ;    Entry type 3 (SSCALL) - Bits 3-35 - Address of routine to call
                                         ;
                                         ;    Entry type 4 (SSCHK) -  Bits 3-35 - Address of routine to call (return
                                         ;                                        +1 if done with test, +2 otherwise)
                                         ;
                                         ;    Entry type 5 (SSJRST) - Bits 3-35 - Table address to continue
                                         ;#***********************************************************************

  044770  261 17 0 00 000000             BEXEC:  RPUT    (0,1,2,3)               ; save AC's

                                         ; Check if end of table

  044774  200 02 0 06 000000                     MOVE    2,(6)                   ; get entry (single step data)
  044775  200 03 0 06 000001                     MOVE    3,1(6)                  ; get entry (EBUF correct data)
  044776  350 00 0 00 000006                     AOS     6                       ; point to next entry
  044777  322 02 0 00 045003                     JUMPE   2,BEXEX1                ; end of table - yes - exit

                                         ; Determine type of entry and dispatch

  045000  135 01 0 00 062471                     LDB     1,[POINT 3,2,2]         ; get entry
                                                 XCT     [JRST   BEXEX1          ; dispatch on it
                                                          JRST   BEXSSS
                                                          JRST   BEXSSC
                                                          JRST   BEXCAL
                                                          JRST   BEXCHK
                                                          JRST   BEXJRS
                                                          JRST   BEXERR
                                                          JRST   BEXERR
                                                          JRST   BEXERR
  045001  256 00 0 01 062724                              JRST   BEXERR](1)

                                         ; Error in Single Step Table

  045002  037 00 0 00 062504             BEXERR: TMSGCD  <Single Step Dispatch Table Error - Entry out of range>
  045003  332 00 0 00 064641'            BEXEX1: SKIPE   SDEBUG                  ; EXEC debug mode?
  045004  037 00 0 00 030242                     PCRL                            ; yes - a final CRLF
  045005  262 17 0 00 000003                     RGET    (3,2,1,0)               ; restore AC's
  045011  263 17 0 00 000000                     RTN                             ; return
```

```

                                         ; Execute single step
```

```


                                      ; Execute single step

045012  135 00 0 00 062520            BEXSSS: LDB     [POINT 12,2,23]         ; get start address
045013  202 00 0 00 044640*                   MOVEM   SNEXT                   ; save it
045014  200 00 0 00 045013*           BEXSSC: MOVE    SNEXT                   ; get next address
045015  202 00 0 00 044551                    MOVEM   SSADDR                  ; save as start address
045016  135 00 0 00 062521                    LDB     [POINT 9,2,11]          ; get # of single steps to do
045017  202 00 0 00 044550                    MOVEM   NSSTEP                  ; save it
045020  135 00 0 00 062522                    LDB     [POINT 12,2,35]         ; get correct end address
045021  202 00 0 00 044552                    MOVEM   SEADDR                  ; save it
045022  202 03 0 00 045142                    MOVEM   3,CEBUF                 ; save correct EBUF data
045023  350 00 0 00 047012                    AOS     TSTSUB                  ; increment test segment
045024  350 00 0 00 000006                    AOS     6                       ; point to next entry

                                      ; Set up start data

045025  336 01 0 00 044462*                   SKIPN   1,SDATA                 ; any data given?
045026  201 01 0 00 000014                    MOVEI   1,MPRUN!4               ; no - set initial data
045027  332 00 0 00 030037                    SKIPE   USER                    ; user mode?
045030  620 01 0 00 000007                    TRZ     1,7                     ; yes - clear PI level bits
045031  660 01 0 00 220010                    TRO     1,TSTEBF!MPRUN!SINCYC
045032  202 01 0 00 045025*                   MOVEM   1,SDATA                 ; save start data
045033  620 01 0 00 200010                    TRZ     1,TSTEBF!MPRUN          ; ensure TSTEBF,MPRUN cleared
045034  260 17 0 00 044670*                   GO      LDCSR                   ; set up to write to LAR

                                      ; Single step NSSTEP times

045035  200 02 0 00 044550                    MOVE    2,NSSTEP                ; get number of times

                                      ; First write the LAR

045036  200 01 0 00 045014*           BEXSSO: MOVE    1,SNEXT                 ; get start address
045037  242 01 0 00 000001                    LSH     1,1                     ; position properly
045040  260 17 0 00 044642*                   GO      LDRAR                   ; load the register

                                      ; Then do the single step

045041  200 01 0 00 045032*                   MOVE    1,SDATA                 ; get start data
045042  260 17 0 00 045034*                   GO      LDCSR                   ; write to CSR

                                      ; Then read the EBUF

045043  260 17 0 00 044671*                   GO      RDEBUF                  ; read it
045044  202 01 0 00 045141                    MOVEM   1,AEBUF                 ; save actual EBUF data
```

```

                                      ; Then read the LAR
```

```

                                       ; Then read the LAR

045045  200 01 0 00 045041*                     MOVE    1,SDATA                 ; get start data
045046  620 01 0 00 200010                      TRZ     1,TSTEBF!MPRUN          ; ensure TSTEBF and MPRUN cleared
045047  660 01 0 00 040000                      TRO     1,SELLAR                ;   and set SELLAR
045050  260 17 0 00 045042*                     GO      LDCSR                   ; write to CSR
045051  260 17 0 00 044675*                     GO      RDLAR                   ; read LAR
045052  242 01 0 00 777777                      LSH     1,-1                    ; position properly
045053  202 01 0 00 045036*                     MOVEM   1,SNEXT                 ; set up new next address
045054  367 02 0 00 045036                      SOJG    2,BEXSSO                ; loop till done
045055  202 01 0 00 044553                      MOVEM   1,SAADDR                ; save actual end address

                                       ; Debug output

045056  332 00 0 00 064641'                     SKIPE   SDEBUG                  ; EXEC debug mode?
                                                GO      [TMSGC  <SSTEP:  #S >   ; yes - print debug
                                                         MOVE   NSSTEP          ;   output and then
                                                         PNTDEC                 ;   continue
                                                         TMSG   <, SADR >
                                                         MOVE   SSADDR
                                                         PNTOCS
                                                         TMSG   <, EADR >
                                                         MOVE   SEADDR
                                                         PNTOCS
                                                         TMSG   <, AADR >
                                                         MOVE   SAADDR
                                                         PNTOCS
                                                         TMSG   <, EBUF >
                                                         MOVE   AEBUF
                                                         PNTHW
045057  260 17 0 00 062740                               RTN]

                                       ; Done - compare end address to expected

045060  402 00 0 00 045143                      SETZM   BFLAG                   ; clear 'incorrect address' flag
045061  200 01 0 00 044553                      MOVE    1,SAADDR                ; save actual end address
045062  312 01 0 00 044552                      CAME    1,SEADDR                ; correct result?
045063  476 00 0 00 045143                      SETOM   BFLAG                   ; no - set 'incorrect address' flag
045064  200 01 0 00 045141                      MOVE    1,AEBUF                 ; get actual EBUF data
045065  316 01 0 00 045142                      CAMN    1,CEBUF                 ; correct result?
045066  350 00 0 17 777774                      AOS     -4(P)                   ; yes - increment return
045067  350 00 0 17 777774                      AOS     -4(P)                   ; set up return
045070  350 00 0 17 777774                      AOS     -4(P)                   ; set up return
045071  262 17 0 00 000003                      RGET    (3,2,1,0)               ; restore AC's
045075  263 17 0 00 000000                      RTN                             ; return +3/+4
```

```

                                       ; Execute a JRST and exit
```

```


                                ; Execute a JRST and exit

045076  550 02 0 00 000002      BEXJRS: HRRZ    2,2                     ; only want the address
045077  332 00 0 00 064641'             SKIPE   SDEBUG                  ; EXEC debug mode?
                                        GO      [TMSGC  <SJRST Adr  >   ; yes - print debug
                                                 MOVE   2               ;   data and continue
                                                 PNTOCS
045100  260 17 0 00 062554                       RTN]
045101  200 06 0 00 000002              MOVE    6,2                     ; get new table address
045102  262 17 0 00 000003              RGET    (3,2,1,0)               ; restore AC's
045106  350 00 0 17 000000              AOS     (P)                     ; set up return
045107  263 17 0 00 000000              RTN                             ; return +2

                                ; Execute a CALL and exit

045110  550 02 0 00 000002      BEXCAL: HRRZ    2,2                     ; only want the address
045111  332 00 0 00 064641'             SKIPE   SDEBUG                  ; EXEC debug mode?
                                        GO      [TMSGC  <SCALL Adr  >   ; yes - print debug
                                                 MOVE   2               ;   data and continue
                                                 PNTOCS
045112  260 17 0 00 062563                       RTN]
045113  261 17 0 00 000004              PUT     4                       ; save another AC
045114  260 17 1 00 000002              GO      @2                      ; call the routine
045115  262 17 0 00 000004              RGET    (4,3,2,1,0)             ; restore AC's
045122  350 00 0 17 000000              AOS     (P)                     ; set up return
045123  263 17 0 00 000000              RTN                             ; return +2

                                ; Execute a CALL and exit (if the return is +1 then exit test)

045124  550 02 0 00 000002      BEXCHK: HRRZ    2,2                     ; only want the address
045125  332 00 0 00 064641'             SKIPE   SDEBUG                  ; EXEC debug mode?
                                        GO      [TMSGC  <SCHK Adr  >    ; yes - print debug
                                                 MOVE   2               ;   data and continue
                                                 PNTOCS
045126  260 17 0 00 062572                       RTN]
045127  261 17 0 00 000004              PUT     4                       ; save another AC
045130  260 17 1 00 000002              GO      @2                      ; call the routine
045131  334 00 0 00 000000              SKIPA                           ; exit test
045132  350 00 0 17 777773              AOS     -5(P)                   ; ok - set up return
045133  262 17 0 00 000004              RGET    (4,3,2,1,0)             ; restore AC's
045140  263 17 0 00 000000              RTN                             ; return +2 or +1

                                ; Miscellaneous


045141  000000  000000          AEBUF:  0                               ; actual EBUF data
045142  000000  000000          CEBUF:  0                               ; correct EBUF data
045143  000000  000000          BFLAG:  0                               ; 'incorrect address' flag
```

```
 8514                                             ;#*********************************************************************
 8515                                             ;* BBPNT - Print out single step data that resulted in an error
 8516                                             ;#*********************************************************************
 8517
 8518
 8519  045144  261 17 0 00 000000          BBPNT:  RPUT    (0,1)
 8520  045146  200 01 0 00 044737*                 MOVE    1,SCOSW                 ; get switches
 8521  045147  603 01 0 00 000200                  TLNE    1,TXTINH                ; inhibit text?
 8522  045150  254 00 0 00 045177                  JRST    BBPN1                   ; yes - print alternate format
 8523
 8524                                             ; Print single step data
 8525
 8526  045151  037 00 0 00 062576                  TMSGC   <Start Addr:  >
 8527  045152  200 00 0 00 044551                  MOVE    SSADDR
 8528  045153  037 04 0 00 000000                  PNT4
 8529  045154  037 00 0 00 062602                  TMSG    <, # Steps:  >
 8530  045155  200 00 0 00 044550                  MOVE    NSSTEP
 8531  045156  037 15 0 00 000000                  PNTDEC
 8532  045157  037 00 0 00 057306                  TMSG    <.>
 8533  045160  336 00 0 00 045143                  SKIPN   BFLAG                   ; incorrect end address
 8534                                              JRST    [TMSGC  <End Addr:  >    ; no - don't print it
 8535                                                       MOVE   SEADDR
 8536                                                       PNT4
 8537  045161  254 00 0 00 062763                           JRST   BBPN0]
 8538  045162  037 00 0 00 062605                  TMSGC   <End Addr - Correct:  >
 8539  045163  200 00 0 00 044552                  MOVE    SEADDR
 8540  045164  037 04 0 00 000000                  PNT4
 8541  045165  037 00 0 00 062612                  TMSGC   <              Actual:  >
 8542  045166  200 00 0 00 044553                  MOVE    SAADDR
 8543  045167  037 04 0 00 000000                  PNT4
 8544
 8545                                             ; Print EBUF correct/actual
 8546
 8547  045170  037 00 0 00 062767          BBPN0:  TMSGC   <EBUF - Correct:  >
 8548  045171  200 00 0 00 045142                  MOVE    CEBUF
 8549  045172  037 13 0 00 000000                  PNTHW
 8550  045173  037 00 0 00 062773                  TMSGC   <         Actual:  >
 8551  045174  200 00 0 00 045141                  MOVE    AEBUF
 8552  045175  037 13 0 00 000000                  PNTHW
 8553  045176  254 00 0 00 045224                  JRST    BBPNX
```

```
 8554
 8555                                             ; Print single step data
 8556
 8557  045177  037 00 0 00 062617          BBPN1:  TMSGC   <Start  >
```

```

                                         ; Print single step data

045177  037 00 0 00 062617               BBPN1:  TMSGC   <Start  >
045200  200 00 0 00 044551                       MOVE    SSADDR
045201  037 04 0 00 000000                       PNT4
045202  037 00 0 00 062621                       TMSG    <, # Steps  >
045203  200 00 0 00 044550                       MOVE    NSSTEP
045204  037 15 0 00 000000                       PNTDEC
045205  037 00 0 00 057306                       TMSG    <.>
045206  336 00 0 00 045143                       SKIPN   BFLAG                   ; incorrect end address
                                                 JRST    [TMSGC  <End  >         ; no - don't print it
                                                          MOVE   SEADDR
                                                          PNT4
045207  254 00 0 00 063001                                JRST   BBPN2]
045210  037 00 0 00 062624                       TMSGC   <End (C):  >
045211  200 00 0 00 044552                       MOVE    SEADDR
045212  037 04 0 00 000000                       PNT4
045213  037 00 0 00 062627                       TMSGC   <    (A):  >
045214  200 00 0 00 044553                       MOVE    SAADDR
045215  037 04 0 00 000000                       PNT4

                                         ; Print EBUF correct/actual

045216  037 00 0 00 063005               BBPN2:  TMSGC   <EBUF (C):  >
045217  200 00 0 00 045142                       MOVE    CEBUF
045220  037 13 0 00 000000                       PNTHW
045221  037 00 0 00 063010                       TMSGC   <     (A):  >
045222  200 00 0 00 045141                       MOVE    AEBUF
045223  037 13 0 00 000000                       PNTHW

                                         ; Return

045224  262 17 0 00 000001               BBPNX:  RGET    (1,0)                   ; restore AC's
045226  263 17 0 00 000000                       RTN
```

```
                                         ;#**************************************************************************
                                         ;* TEXEC - Test execute routine
                                         ;
```

```
;#***********************************************************************
;* TEXEC - Test execute routine
;
; This routine starts up the port and reads CSR register  to find
; out the results of tests performed by microcode.  It  may  also
; stop the port to read the EBUF and verify its contents.
;
; This  routine is given a pointer to a 'Test Execute Table'.  It
; then executes one entry and returns +1,+2,+3 or +4 depending on
; the type of entry and the result.
;
; Argument:     AC6 - Address of 'Test Execute Table'
;
; Return:       +1 - End of table was seen.
;               +2 - A routine was called (just do next entry).
;               +3 - Error return.
;               +4 - Success return.
;
;               TADDR - Initial starting address
;               TADDRF - Final address
;               TCSRF - Final CSR data
;               TTIME - Amount of time to let port run
;               TEBUFC - Correct EBUF data
;               TEBUFA - Actual EBUF data
;               TCOMP - Test completion flag
;                         0 - no - bit 12 never set in CSR
;                        -1 - yes - bit 12 set in CSR
;               TFAIL - Test failed flag
;                         0 - no - bit 13 set in CSR
;                        -1 - yes - bit 13 never set in CSR
;
;               AC6 - Updated to point to the next table location
;
; Test Execute Table Format:
;
;               Entry
;               Entry
;               ...
;               TLAST           ; last entry (zero)
;
; 'Entry' is of the form:
;
; Bits 0-3 - Type of entry - 0 - Done with table              (TLAST)
;                            1 - Start port (start addr given)(TSTART)
;                            2 - Continue                     (TCONT)
;                            3 - Call routine                 (TCALL)
;                            4 - Call routine (test exit)     (TCALLC)
;                            5 - Check for test segment done  (TCHECK)
;                            6 - Select new table address     (TJRST)
;                            7 - Exit test (if error flg set)(TEXIT)
;                           10 - Soft start port (no IPACLR done)(TSSTAR)
;
```

```
; Bits 4-35 - Defined per entry type.
;
;    Entry type 0 (TLAST) -  All zero - indicates end of table
```

```
                                        ; Bits 4-35 - Defined per entry type.
                                        ;
                                        ;   Entry type 0 (TLAST) -  All zero - indicates end of table
                                        ;
                                        ;   Entry type 1 (TSTART) - Bits 4-35 - CRAM address to start at
                                        ;
                                        ;   Entry type 2 (TCONT) -  Bits 4-35 - Unused
                                        ;
                                        ;   Entry type 3 (TCALL) -  Bits 4-35 - Address of routine to call
                                        ;
                                        ;   Entry type 4 (TCALLC) - Bits 4-35 - Address of routine to call
                                        ;
                                        ;   Entry type 5 (TCHECK) - Bits 4-17 - Time to wait for port to complete
                                        ;                           Bits 18-35 - Address of correct EBUF data
                                        ;
                                        ;   Entry type 6 (TJRST) -  Bits 4-35 - Table address to continue
                                        ;
                                        ;   Entry type 7 (TEXIT) - Bits 4-35 - Unused
                                        ;
                                        ;   Entry type 10 (TSSTAR) - Bits 4-35 - CRAM address to start at
                                        ;#*****************************************************************

045227  261 17 0 00 000000              TEXEC:  RPUT    (0,1,2)                 ; save some AC's

                                        ; Check if end of table

045232  200 02 0 06 000000                      MOVE    2,(6)                   ; get entry
045233  350 00 0 00 000006                      AOS     6                       ; point to next entry
045234  322 02 0 00 045240                      JUMPE   2,TEXEX1                ; end of table - yes - exit

                                        ; Determine type of entry and dispatch

045235  135 01 0 00 063013                      LDB     1,[POINT 4,2,3]         ; get entry
                                                XCT     [JRST   TEXERR          ; dispatch on it
                                                         JRST   TEXSTR
                                                         JRST   TEXCON
                                                         JRST   TEXCAL
                                                         JRST   TEXCAC
                                                         JRST   TEXCHK
                                                         JRST   TEXJRS
                                                         JRST   TEXEXI
                                                         JRST   TEXSST
045236  256 00 0 01 063014                               JRST   TEXERR](1)
```

```
                                        ; Error in Test Execute Table

045237  037 00 0 00 063026              TEXERR: TMSGCD  <Test Execute Dispatch Table Error - Entry out of range>
```

```
                                          ; Error in Test Execute Table

045237  037 00 0 00 063026                TEXERR: TMSGCD  <Test Execute Dispatch Table Error - Entry out of range>
045240  332 00 0 00 064641'               TEXEX1: SKIPE   SDEBUG                    ; EXEC debug mode?
045241  037 00 0 00 030242                        PCRL                              ; yes - a final CRLF
045242  262 17 0 00 000002                        RGET    (2,1,0)                   ; restore AC's
045245  263 17 0 00 000000                        RTN                               ; return

                                          ; Start/continue the port

045246  135 01 0 00 063042                TEXSST: LDB     1,[POINT 32,2,35]         ; get start address
045247  301 01 0 00 007777                        CAIL    1,7777                    ; default address selected?
045250  200 01 0 00 045430                        MOVE    1,TADDR                   ; yes - get it
045251  202 01 0 00 045053*                       MOVEM   1,SNEXT                   ; save it
045252  254 00 0 00 045260                        JRST    TEXCO0                    ; continue
045253  135 01 0 00 063042                TEXSTR: LDB     1,[POINT 32,2,35]         ; get start address
045254  301 01 0 00 007777                        CAIL    1,7777                    ; default address selected?
045255  200 01 0 00 045430                        MOVE    1,TADDR                   ; yes - get it
045256  202 01 0 00 045251*                       MOVEM   1,SNEXT                   ; save it
045257  260 17 0 00 044664*               TEXCON: GO      IPACLR                    ; do a 'port clear'
045260  200 01 0 00 045256*               TEXCO0: MOVE    1,SNEXT                   ; get next address
045261  202 01 0 00 045430                        MOVEM   1,TADDR                   ; save as start address
045262  242 01 0 00 000001                        LSH     1,1                       ; position properly
045263  260 17 0 00 045040*                       GO      LDRAR                     ; load the register
045264  336 01 0 00 045045*                       SKIPN   1,SDATA                   ; any data given?
045265  201 01 0 00 000014                        MOVEI   1,MPRUN!4                 ; no - set initial data
045266  332 00 0 00 030037                        SKIPE   USER                      ; user mode?
045267  620 01 0 00 000007                        TRZ     1,7                       ; yes - clear PI level bits
045270  660 01 0 00 040010                        TRO     1,SELLAR!MPRUN            ; ensure SELLAR,MPRUN are set
045271  202 01 0 00 045264*                       MOVEM   1,SDATA                   ; save start data
045272  260 17 0 00 045050*                       GO      LDCSR                     ; write to CSR
045273  332 00 0 00 064641'                       SKIPE   SDEBUG                    ; EXEC debug mode?
                                                  GO      [TMSGC  <TSTART/CONT  Adr  > ; yes - print debug
                                                           MOVE    SNEXT              ;   data and continue
                                                           PNTOCS
                                                           RTN]
045274  260 17 0 00 063050
045275  262 17 0 00 000002                        RGET    (2,1,0)                   ; restore AC's
045300  350 00 0 17 000000                        AOS     (P)                       ; set up return
045301  263 17 0 00 000000                        RTN                               ; return +3/+4
```

```
                                          ; Execute a JRST and exit

045302  550 02 0 00 000002                TEXJRS: HRRZ    2,2                       ; only want the address
```

```

                                        ; Execute a JRST and exit

045302  550 02 0 00 000002              TEXJRS: HRRZ    2,2                     ; only want the address
045303  332 00 0 00 064641'                     SKIPE   SDEBUG                  ; EXEC debug mode?
                                                GO      [TMSGC  <TJRST Adr  >   ; yes - print debug
                                                         MOVE   2               ;   data and continue
                                                         PNTOCS
045304  260 17 0 00 063057                               RTN]
045305  200 06 0 00 000002                      MOVE    6,2                     ; get new table address
045306  262 17 0 00 000002                      RGET    (2,1,0)                 ; restore AC's
045311  350 00 0 17 000000                      AOS     (P)                     ; set up return
045312  263 17 0 00 000000                      RTN                             ; return +2

                                        ; Execute a CALL and exit

045313  550 02 0 00 000002              TEXCAL: HRRZ    2,2                     ; only want the address
045314  332 00 0 00 064641'                     SKIPE   SDEBUG                  ; EXEC debug mode?
                                                GO      [TMSGC  <TCALL Adr  >   ; yes - print debug
                                                         MOVE   2               ;   data and continue
                                                         PNTOCS
045315  260 17 0 00 063066                               RTN]
045316  261 17 0 00 000003                      RPUT    (3,4)                   ; save some AC's
045320  260 17 1 00 000002                      GO      @2                      ; call the routine
045321  262 17 0 00 000004                      RGET    (4,3,2,1,0)             ; restore AC's
045326  350 00 0 17 000000                      AOS     (P)                     ; set up return
045327  263 17 0 00 000000                      RTN                             ; return +2

                                        ; Execute a CALL and check if test done and exit

045330  550 02 0 00 000002              TEXCAC: HRRZ    2,2                     ; only want the address
045331  332 00 0 00 064641'                     SKIPE   SDEBUG                  ; EXEC debug mode?
                                                GO      [TMSGC  <TCALLC Adr  >  ; yes - print debug
                                                         MOVE   2               ;   data and continue
                                                         PNTOCS
045332  260 17 0 00 063075                               RTN]
045333  261 17 0 00 000003                      RPUT    (3,4)                   ; save some AC's
045335  260 17 1 00 000002                      GO      @2                      ; call the routine
045336  334 00 0 00 000000                      SKIPA                           ; exit test
045337  350 00 0 17 777773                      AOS     -5(P)                   ; set up return
045340  262 17 0 00 000004                      RGET    (4,3,2,1,0)             ; restore AC's
045345  263 17 0 00 000000                      RTN                             ; return +2
```

```

                                        ; Exit test (if error flag is set)

045346  332 00 0 00 064641'             TEXEXI: SKIPE   SDEBUG                  ; EXEC debug mode?
```

```
                                        ; Exit test (if error flag is set)

045346  332 00 0 00 064641'     TEXEXI: SKIPE   SDEBUG                          ; EXEC debug mode?
                                        GO      [TMSGC  <TEXIT>                 ; yes - print debug
045347  260 17 0 00 063103                       RTN]                           ; data and continue
045350  336 00 0 00 000015              SKIPN   ERFLG                           ; any errors yet?
045351  350 00 0 17 777775              AOS     -3(P)                           ; no - not done with test yet
045352  262 17 0 00 000002              RGET    (2,1,0)                         ; restore AC's
045355  263 17 0 00 000000              RTN                                     ; return +2 or +1

                                ; Check for test segment done

045356  350 00 0 00 047012      TEXCHK: AOS     TSTSUB                          ; increment test segment
045357  402 00 0 00 045436              SETZM   TCOMP                           ; clear completed flag
045360  402 00 0 00 045437              SETZM   TFAIL                           ; clear failed flag
045361  135 00 0 00 063105              LDB     [POINT 14,2,17]                 ; get time to wait
045362  260 17 0 00 043121              GO      ODELAY                          ; wait
045363  260 17 0 00 044665*             GO      RDCSR                           ; read CSR
045364  474 15 0 00 000000              SETO    ERFLG,                          ; error accessing CSR
045365  202 01 0 00 045432              MOVEM   1,TCSRF                         ; save final CSR data
045366  603 01 0 00 000020              TLNE    1,(1B13)                        ; bit 13 set?
045367  476 00 0 00 045437              SETOM   TFAIL                           ; yes - test failed
045370  603 01 0 00 000040              TLNE    1,(1B12)                        ; bit 12 set?
045371  476 00 0 00 045436              SETOM   TCOMP                           ; yes - test never completed
045372  332 00 0 00 045437              SKIPE   TFAIL                           ; test failed?
045373  474 15 0 00 000000              SETO    ERFLG,                          ; yes - set error flag
045374  336 00 0 00 045436              SKIPN   TCOMP                           ; test completed?
045375  474 15 0 00 000000              SETO    ERFLG,                          ; no - set error flag

                                ; Now read LAR to see where stopped, and read EBUF also

045376  201 01 0 00 200000              MOVEI   1,TSTEBF                        ; set up CSR data
045377  260 17 0 00 045272*             GO      LDCSR                           ; write it
045400  260 17 0 00 045043*             GO      RDEBUF                          ; read EBUF
045401  202 01 0 00 045435              MOVEM   1,TEBUFA                        ; save it (actual data)
045402  202 01 0 00 045434              MOVEM   1,TEBUFC                        ; save it (correct data)
045403  550 01 0 00 000002              HRRZ    1,2                             ; get address of correct data
045404  322 01 0 00 045411              JUMPE   1,TEXCHO                        ; unused? - yes - continue
045405  200 01 0 01 000000              MOVE    1,(1)                           ; get correct data
045406  202 01 0 00 045434              MOVEM   1,TEBUFC                        ; save it (correct data)
045407  312 01 0 00 045435              CAME    1,TEBUFA                        ; EBUF data correct?
045410  474 15 0 00 000000              SETO    ERFLG,                          ; no - set error flag
045411  201 01 0 00 040000      TEXCHO: MOVEI   1,SELLAR                        ; set up CSR data
045412  260 17 0 00 045377*             GO      LDCSR                           ; write it
045413  260 17 0 00 045051*             GO      RDLAR                           ; read LAR
045414  242 01 0 00 777777              LSH     1,-1                            ; position correctly
045415  202 01 0 00 045431              MOVEM   1,TADDRF                        ; save it
```

```
                                ; And exit +3 or +4

045416  332 00 0 00 064641'             SKIPE   SDEBUG                          ; EXEC debug mode?
```

```

                                    ; And exit +3 or +4

                                            SKIPE   SDEBUG                  ; EXEC debug mode?
045416  332 00 0 00 064641'                 GO      [TMSGC  <TCHECK CSR >   ; yes - print debug
                                                     MOVE   TCSRF           ;   data and continue
                                                     PNTHW
                                                     TMSG   < EBUF >
                                                     MOVE   TEBUFA
                                                     PNTOCS
                                                     TMSG   < ADR >
                                                     MOVE   TADDRF
                                                     PNTOCS
045417  260 17 0 00 063115                           RTN]
045420  336 00 0 00 000015                  SKIPN   ERFLG                   ; error flag set?
045421  350 00 0 17 777775                  AOS     -3(P)                   ; no - set up good return
045422  350 00 0 17 777775                  AOS     -3(P)                   ; set up return
045423  350 00 0 17 777775                  AOS     -3(P)                   ; set up return
045424  262 17 0 00 000002                  RGET    (2,1,0)                 ; restore AC's
045427  263 17 0 00 000000                  RTN                             ; return

                                    ; Miscellaneous

045430  000000  000000              TADDR:  0                               ; initial starting address
045431  000000  000000              TADDRF: 0                               ; final address
045432  000000  000000              TCSRF:  0                               ; final CSR data
045433  000000  000000              TTIME:  0                               ; amount of time to let port run
045434  000000  000000              TEBUFC: 0                               ; correct EBUF data
045435  000000  000000              TEBUFA: 0                               ; actual EBUF data
045436  000000  000000              TCOMP:  0                               ; test completion flag (0-no,1-yes
                                                                            ;   bit 12 did set in CSR)
045437  000000  000000              TFAIL:  0                               ; test failed flag (0-no,1-yes
                                                                            ;   bit 13 never set in CSR)



                                    ;#*********************************************************************
                                    ;* TTPNT - Print out error data
                                    ;#*********************************************************************

045440  261 17 0 00 000000          TTPNT:  RPUT    (0,1)                   ; save AC's
045442  200 01 0 00 045146*                 MOVE    1,SCOSW                 ; get switches
045443  037 00 0 00 062576                  TMSGC   <Start Addr:  >         ; print start address
045444  200 00 0 00 045430                  MOVE    TADDR
045445  037 04 0 00 000000                  PNT4
```

```

                                    ; Check test handshake

045446  332 00 0 00 045436                  SKIPE   TCOMP                   ; test completed?
```

```


                                           ; Check test handshake

 045446  332 00 0 00 045436                        SKIPE   TCOMP                           ; test completed?
 045447  254 00 0 00 045456                        JRST    TTPN0                           ; yes - see how it failed
 045450  332 00 0 00 045437                        SKIPE   TFAIL                           ; handshake failed?
 045451  254 00 0 00 045456                        JRST    TTPN0                           ; no - continue
 045452  037 00 0 00 063127                        TMSGC   <Test handshake unsuccessful - final address:  >
 045453  200 00 0 00 045431                        MOVE    TADDRF                          ; get final address
 045454  037 04 0 00 000000                        PNT4                                    ; print it
 045455  254 00 0 00 045463                        JRST    TTPN1                           ; exit

                                           ; Check test completion

 045456  336 00 0 00 045437                TTPN0:  SKIPN   TFAIL                           ; test really did fail?
 045457  254 00 0 00 045463                        JRST    TTPN1                           ; no - exit
 045460  037 00 0 00 063141                        TMSGC   <Test did not complete successfully - final address:  >
 045461  200 00 0 00 045431                        MOVE    TADDRF                          ; get final address
 045462  037 04 0 00 000000                        PNT4                                    ; print it

                                           ; Print final CSR data

 045463  603 01 0 00 000200                TTPN1:  TLNE    1,TXTINH                        ; full printout?
 045464  254 00 0 00 045470                        JRST    TTPN2                           ; no - continue
 045465  037 00 0 00 063155                        TMSGC   <Final CSR data:  >
 045466  200 00 0 00 045432                        MOVE    TCSRF                           ; get data
 045467  037 13 0 00 000000                        PNTHW                                   ; print it

                                           ; Finally check EBUF data

 045470  200 00 0 00 045434                TTPN2:  MOVE    TEBUFC                          ; get correct EBUF data
 045471  316 00 0 00 045435                        CAMN    TEBUFA                          ; same as actual?
 045472  254 00 0 00 045507                        JRST    TTPNX                           ; yes - exit
 045473  603 01 0 00 000200                        TLNE    1,TXTINH                        ; full printout?
 045474  037 00 0 00 063005                        TMSGC   <EBUF (C):  >                   ; no
 045475  607 01 0 00 000200                        TLNN    1,TXTINH                        ; full printout?
 045476  037 00 0 00 062767                        TMSGC   <EBUF - Correct:  >             ; yes
 045477  200 00 0 00 045434                        MOVE    TEBUFC
 045500  037 13 0 00 000000                        PNTHW
 045501  603 01 0 00 000200                        TLNE    1,TXTINH                        ; full printout?
 045502  037 00 0 00 063010                        TMSGC   <     (A):  >                   ; no
 045503  607 01 0 00 000200                        TLNN    1,TXTINH                        ; full printout?
 045504  037 00 0 00 062773                        TMSGC   <          Actual:  >           ; yes
 045505  200 00 0 00 045435                        MOVE    TEBUFA
 045506  037 13 0 00 000000                        PNTHW

                                           ; Done - exit

 045507  262 17 0 00 000001                TTPNX:  RGET    (1,0)                           ; restore AC's
 045511  263 17 0 00 000000                        RTN                                     ; return
```

```
                                           ;*************************************************************************
                                           ;* CEXEC - Data test execute routine
                                           ;
```

```

;*************************************************************************
;* CEXEC - Data test execute routine
;
; This routine is used to execute a CBUS data transfer type test.  It
; allows data transfers to be set up, specifying buffer  lengths  and
; data pattern.  It can set up and start a data transfer, and inspect
; the result, and do data compare of the resulting buffer.
;
; This routine is given a pointer to a 'Data Test Execute Table'.  It
; then executes one entry and returns +1,+2,+3 or +4 depending on the
; type of entry and the result.
;
; Argument:     AC6 - Address of 'Data Test Execute Table'
;
; Return:       +1 - End of table was seen.
;               +2 - A routine was called (just do next entry).
;               +3 - Error return.
;               +4 - Success return.
;
;               QADDR - Initial starting address
;               QADDRF - Final address
;               CTIME - Amount of time to let port run
;               CEBUFC - Correct EBUF data
;               CEBUFA - Actual EBUF data
;               CPAT - Data pattern used
;               CLEN - Buffer length
;               CTRAN - Type of transfer 0-none,1-rd,2-wr,3-rd/wr
;               CDERR - Data compare error - 0-no,-1-yes
;               CSERR - Status error - 0-none,xx-error bits
;
;               AC6 - Updated to point to the next table location
;
; Test Execute Table Format:
;
;               Entry
;                ...
;               CLAST           ; last entry (zero)
;
; 'Entry' is of the form:
;
; Bits 0-3 - Type of entry
;    0 - Done with table (CLAST)        6 - Call routine (test exit) (CCALLC)
;    1 - Set up read DT  (CSETRD)       7 - Wait for dt completion   (CWAIi)
;    2 - Set up write DT (CSETWR)      10 - Compare data buffer      (CCOMP)
;    3 - Set up rd/wr DT (CSETRW)      11 - Select new table address (CJRST)
;    4 - Start port      (CSTART)      12 - Exit test (if erflg set) (CEXIT)
;    5 - Call routine    (CCALL)
```

```
; Bits 4-35 - Defined per entry type.
;
;     Entry type 0 (CLAST) -  All zero - indicates end of table
```

```
                                  ; Bits 4-35 - Defined per entry type.
                                  ;
                                  ;    Entry type 0 (CLAST) -  All zero - indicates end of table
                                  ;
                                  ;    Entry type 1 (CSETWR) - Bits 4-17 - data pattern (0=default)
                                  ;                                18-35 - buffer length
                                  ;
                                  ;    Entry type 2 (CSETRD) - Bits 4-17 - data pattern (0=default)
                                  ;                                18-35 - buffer length
                                  ;
                                  ;    Entry type 3 (CSETRW) - Bits 4-17 - data pattern (0=default)
                                  ;                                18-35 - buffer length
                                  ;
                                  ;    Entry type 4 (CSTART) - Bits 4-35 - CRAM address to start at
                                  ;
                                  ;    Entry type 5 (CCALL) -  Bits 4-35 - Address of routine to call
                                  ;
                                  ;    Entry type 6 (CCALLC) - Bits 4-35 - Address of routine to call
                                  ;
                                  ;    Entry type 7 (CWAIT) - Bits 4-17 - time to wait (in milliseconds)
                                  ;
                                  ;    Entry type 10 (CCOMP) - Bits 4-35 - Unused
                                  ;
                                  ;    Entry type 11 (CJRST) -  Bits 4-35 - Table address to continue
                                  ;
                                  ;    Entry type 12 (CEXIT) - Bits 4-35 - Unused
                                  ;#*******************************************************************

045512  261 17 0 00 000000        CEXEC:  RPUT    (0,1,2)                  ; save some AC's

                                  ; Check if end of table

045515  200 02 0 06 000000                MOVE    2,(6)                    ; get entry
045516  350 00 0 00 000006                AOS     6                        ; point to next entry
045517  322 02 0 00 045523                JUMPE   2,CEXEX1                 ; end of table - yes - exit
```

```

                                  ; Determine type of entry and dispatch

045520  135 01 0 00 063013                LDB     1,[POINT 4,2,3]          ; get entry
```

```
                                   ; Determine type of entry and dispatch

045520  135 01 0 00 063013                 LDB     1,[POINT 4,2,3]         ; get entry
                                           XCT     [JRST   CEXERR          ; dispatch on it
                                                    JRST   CEXSRD
                                                    JRST   CEXSWR
                                                    JRST   CEXSRW
                                                    JRST   CEXSTR
                                                    JRST   CEXCAL
                                                    JRST   CEXCAC
                                                    JRST   CEXWAI
                                                    JRST   CEXCOM
                                                    JRST   CEXJRS
                                                    JRST   CEXEXI
                                                    JRST   CEXERR
045521  256 00 0 01 063161                          JRST   CEXERR](1)

                                   ; Error in Test Execute Table

045522  037 00 0 00 063176         CEXERR: TMSGCD  <Data Test Execute Dispatch Table Error - Entry out of range>
045523  332 00 0 00 064641'        CEXEX1: SKIPE   SDEBUG                  ; EXEC debug mode?
045524  037 00 0 00 030242                 PCRL                            ; yes - a final CRLF
045525  262 17 0 00 000002                 RGET    (2,1,0)                 ; restore AC's
045530  263 17 0 00 000000                 RTN                             ; return

                                   ; Set up a read data transfer

045531  550 00 0 00 000002         CEXSRD: HRRZ    2                       ; get transfer length
045532  332 00 0 00 000000                 SKIPE                           ; zero?
045533  202 00 0 00 046036                 MOVEM   CLEN                    ; no - save it
045534  135 00 0 00 063105                 LDB     [POINT 14,2,17]         ; get data pattern
045535  332 00 0 00 000000                 SKIPE                           ; zero?
045536  202 00 0 00 046035                 MOVEM   CPAT                    ; no - save it
045537  201 00 0 00 000001                 MOVEI   1                       ; get type of transfer (read)
045540  202 00 0 00 046037                 MOVEM   CTRAN                   ; save it
045541  201 01 0 00 005000         CSETCC: MOVEI   1,5000                  ; get start address
045542  242 01 0 00 000001                 LSH     1,1                     ; position correctly
045543  260 17 0 00 045263*                GO      LDRAR                   ; load the register
045544  201 01 0 00 000010                 MOVEI   1,MPRUN                 ; get 'MPRUN' bit
045545  260 17 0 00 045412*                GO      LDCSR                   ; start up the port
045546  200 01 0 00 063213                 MOVE    1,[400000,,1]           ; get fill pattern
045547  202 01 0 00 050000                 MOVEM   1,BUFF                  ; put fill pattern throughout
045550  200 01 0 00 063214                 MOVE    1,[BUFF,,BUFF+1]        ;   entire buffer area
045551  251 01 0 00 050777                 BLT     1,BUFF+^D511            ;   ...
045552  200 14 0 00 046035                 MOVE    PAT,CPAT                ; get data pattern
045553  200 01 0 00 046037                 MOVE    1,CTRAN                 ; get transfer type
045554  200 02 0 00 046036                 MOVE    2,CLEN                  ; get transfer length
045555  306 01 0 00 000003                 CAIN    1,3                     ; read/write?
045556  242 02 0 00 777777                 LSH     2,-1                    ; yes - divide by 2
045557  260 17 0 00 042521*                GO      BUFGEN                  ; generate buffer contents
```

```
045560  200 01 0 00 061774                 MOVE    1,[CBUFSZ,,CBUF]        ; get pointer to control buffer
045561  260 17 0 00 042523*                GO      CHINIT                  ; initialize software
045562  551 01 0 00 050000                 HRRZI   1,BUFF                  ; buffer address
```

```
045560  200 01 0 00 061774            MOVE    1,[CBUFSZ,,CBUF]          ; get pointer to control buffer
045561  260 17 0 00 042523*           GO      CHINIT                    ; initialize software
045562  551 01 0 00 050000            HRRZI   1,BUFF                    ; buffer address
045563  200 02 0 00 046036            MOVE    2,CLEN                    ; word count
045564  200 03 0 00 046037            MOVE    3,CTRAN                   ; get transfer type
045565  306 03 0 00 000003            CAIN    3,3                       ; read/write?
045566  242 02 0 00 777777            LSH     2,-1                      ; yes - divide by 2
045567  400 03 0 00 000000            SETZ    3,                        ; clear zero fill specifier
045570  260 17 0 00 042527*           GO      GENCCW                    ; generate a CCW list
045571  332 00 0 00 064641'           SKIPE   SDEBUG                    ; EXEC debug mode?
                                      GO      [TMSGC  <CSET>            ; yes - print debug
                                                MOVE    1,CTRAN         ;   data and continue
                                                XCT     [TMSG   <RD>
                                                         TMSG   <WR>
                                                         TMSG   <RW>](1)
                                                TMSG    <  Pat=>
                                                MOVE    CPAT
                                                PNTOCS
                                                TMSG    <  Len=>
                                                MOVE    CLEN
                                                PNTOCS
045572  260 17 0 00 063231                      RTN]
045573  350 00 0 17 777775            AOS     -3(P)                     ; set up proper return
045574  262 17 0 00 000002            RGET    (2,1,0)                   ; restore AC's
045577  263 17 0 00 000000            RTN                               ; return

                              ; Set up a write data transfer

045600  550 00 0 00 000002    CEXSWR: HRRZ    2                         ; get transfer length
045601  332 00 0 00 000000            SKIPE                             ; zero?
045602  202 00 0 00 046036            MOVEM   CLEN                      ; no - save it
045603  135 00 0 00 063105            LDB     [POINT 14,2,17]           ; get data pattern
045604  332 00 0 00 000000            SKIPE                             ; zero?
045605  202 00 0 00 046035            MOVEM   CPAT                      ; no - save it
045606  201 00 0 00 000002            MOVEI   2                         ; get type of transfer (write)
045607  202 00 0 00 046037            MOVEM   CTRAN                     ; save it
045610  254 00 0 00 045541            JRST    CSETCC                    ; common code

                              ; Set up a read/write data transfer

045611  550 00 0 00 000002    CEXSRW: HRRZ    2                         ; get transfer length
045612  332 00 0 00 000000            SKIPE                             ; zero?
045613  202 00 0 00 046036            MOVEM   CLEN                      ; no - save it
045614  135 00 0 00 063105            LDB     [POINT 14,2,17]           ; get data pattern
045615  332 00 0 00 000000            SKIPE                             ; zero?
045616  202 00 0 00 046035            MOVEM   CPAT                      ; no - save it
045617  201 00 0 00 000003            MOVEI   3                         ; get type of transfer (read/write)
045620  202 00 0 00 046037            MOVEM   CTRAN                     ; save it
045621  200 14 0 00 046035            MOVE    PAT,CPAT                  ; get data pattern
045622  254 00 0 00 045541            JRST    CSETCC                    ; common code
```

```
                                       ; Start the port

045623  260 17 0 00 045257*            CEXSTR: GO      IPACLR                  ; do a 'port clear'
045624  135 01 0 00 063042                     LDB     1,[POINT 32,2,35]       ; get start address
045625  301 01 0 00 007777                     CAIL    1,7777                  ; default address selected?
045626  200 01 0 00 046025                     MOVE    1,QADDR                 ; yes - get it
045627  202 01 0 00 045260*                    MOVEM   1,SNEXT                 ; save it
045630  202 01 0 00 046025                     MOVEM   1,QADDR                 ; save as start address
045631  242 01 0 00 000001                     LSH     1,1                     ; position properly
045632  260 17 0 00 045543*                    GO      LDRAR                   ; load the register
045633  336 01 0 00 045271*                    SKIPN   1,SDATA                 ; any data given?
045634  201 01 0 00 000014                     MOVEI   1,MPRUN!4               ; no - set initial data
045635  332 00 0 00 030037                     SKIPE   USER                    ; user mode?
045636  620 01 0 00 000007                     TRZ     1,7                     ; yes - clear PI level bits
045637  660 01 0 00 040210                     TRO     1,SELLAR!MPRUN!RESQAV   ; ensure SELLAR,MPRUN,RESQAV are set
045640  202 01 0 00 045633*                    MOVEM   1,SDATA                 ; save start data
045641  200 01 0 00 045640*                    MOVE    1,SDATA                 ; get start data
045642  260 17 0 00 045545*                    GO      LDCSR                   ; write to CSR
045643  332 00 0 00 064641'                    SKIPE   SDEBUG                  ; EXEC debug mode?
                                               GO      [TMSGC  <CSTART  Adr  > ; yes - print debug
                                                        MOVE    2              ;   data and continue
                                                        PNTOCS
045644  260 17 0 00 063247                              RTN]
045645  350 00 0 17 777775                     AOS     -3(P)                   ; set up return
045646  262 17 0 00 000002                     RGET    (2,1,0)                 ; restore AC's
045651  263 17 0 00 000000                     RTN                             ; return +3/+4

                                       ; Execute a JRST and exit

045652  550 02 0 00 000002             CEXJRS: HRRZ    2,2                     ; only want the address
045653  332 00 0 00 064641'                    SKIPE   SDEBUG                  ; EXEC debug mode?
                                               GO      [TMSGC  <CJRST Adr  >   ; yes - print debug
                                                        MOVE    2              ;   data and continue
                                                        PNTOCS
045654  260 17 0 00 063256                              RTN]
045655  200 06 0 00 000002                     MOVE    6,2                     ; get new table address
045656  262 17 0 00 000002                     RGET    (2,1,0)                 ; restore AC's
045661  350 00 0 17 000000                     AOS     (P)                     ; set up return
045662  263 17 0 00 000000                     RTN                             ; return +2
```

```
                                       ; Execute a CALL and exit

045663  550 02 0 00 000002             CEXCAL: HRRZ    2,2                     ; only want the address
045664  332 00 0 00 064641'                    SKIPE   SDEBUG                  ; EXEC debug mode?
```

```
                                        ; Execute a CALL and exit

045663  550 02 0 00 000002      CEXCAL: HRRZ    2,2                         ; only want the address
045664  332 00 0 00 064641'             SKIPE   SDEBUG                      ; EXEC debug mode?
                                        GO      [TMSGC  <CCALL Adr  >       ; yes - print debug
                                                 MOVE   2                   ;   data and continue
                                                 PNTOCS
045665  260 17 0 00 063265                       RTN]
045666  261 17 0 00 000003              RPUT    (3,4)                       ; save some AC's
045670  260 17 1 00 000002              GO      @2                          ; call the routine
045671  262 17 0 00 000004              RGET    (4,3,2,1,0)                 ; restore AC's
045676  350 00 0 17 000000              AOS     (P)                         ; set up return
045677  263 17 0 00 000000              RTN                                 ; return +2

                                        ; Execute a CALL and check if test done and exit

045700  550 02 0 00 000002      CEXCAC: HRRZ    2,2                         ; only want the address
045701  332 00 0 00 064641'             SKIPE   SDEBUG                      ; EXEC debug mode?
                                        GO      [TMSGC  <CCALLC Adr  >      ; yes - print debug
                                                 MOVE   2                   ;   data and continue
                                                 PNTOCS
045702  260 17 0 00 063274                       RTN]
045703  261 17 0 00 000003              RPUT    (3,4)                       ; save some AC's
045705  260 17 1 00 000002              GO      @2                          ; call the routine
045706  334 00 0 00 000000              SKIPA                               ; exit test
045707  350 00 0 17 777773              AOS     -5(P)                       ; set up return
045710  262 17 0 00 000004              RGET    (4,3,2,1,0)                 ; restore AC's
045715  263 17 0 00 000000              RTN                                 ; return +2

                                        ; Exit test (if error flag is set)

045716  332 00 0 00 064641'     CEXEXI: SKIPE   SDEBUG                      ; EXEC debug mode?
                                        GO      [TMSGC  <CEXIT>             ; yes - print debug
045717  260 17 0 00 063302                       RTN]                       ; data and continue
045720  336 00 0 00 000015              SKIPN   ERFLG                       ; any errors yet?
045721  350 00 0 17 777775              AOS     -3(P)                       ; no - not done with test yet
045722  262 17 0 00 000002              RGET    (2,1,0)                     ; restore AC's
045725  263 17 0 00 000000              RTN                                 ; return +2 or +1
```

```
                                        ; Wait for data transfer to complete

045726  350 00 0 00 047012      CEXWAI: AOS     TSTSUB                      ; increment test segment
                                        HRRZ    2                           ; get EBUF address
```

```

                                 ; Wait for data transfer to complete

045726  350 00 0 00 047012       CEXWAI: AOS     TSTSUB                  ; increment test segment
045727  550 00 0 00 000002               HRRZ    2                       ; get EBUF address
045730  202 00 0 00 064573'              MOVEM   CEBUAD#                 ; save it
045731  135 00 0 00 063105               LDB     [POINT 14,2,17]         ; get time to wait
045732  260 17 0 00 043156               GO      DDELAY                  ; wait (while read buffer/checking CSR)
045733  260 17 0 00 045363*              GO      RDCSR                   ; read CSR
045734  254 00 0 00 045771               JRST    CEXWA3                  ; error accessing CSR
045735  202 01 0 00 046027               MOVEM   1,CCSR                  ; save CSR data

                                 ; Get and verify EBUF data (if any specified)

045736  201 01 0 00 200000               MOVEI   1,TSTEBF                ; set up to read EBUF
045737  260 17 0 00 045642*              GO      LDCSR                   ; write CSR
045740  260 17 0 00 045400*              GO      RDEBUF                  ; read EBUF
045741  202 01 0 00 046032               MOVEM   1,CEBUFA                ; save it (actual)
045742  202 01 0 00 046031               MOVEM   1,CEBUFC                ; save it (correct)
045743  336 01 0 00 064573'              SKIPN   1,CEBUAD                ; get address of correct EBUF data
045744  254 00 0 00 045747               JRST    CEXWA2                  ; none - continue
045745  200 01 0 01 000000               MOVE    1,(1)                   ; get correct EBUF data
045746  202 01 0 00 046031               MOVEM   1,CEBUFC                ; save it
045747  200 01 0 00 046032       CEXWA2: MOVE    1,CEBUFA                ; get actual data
045750  312 01 0 00 046031               CAME    1,CEBUFC                ; error?
045751  474 15 0 00 000000               SETO    ERFLG,                  ; yes - set error flag

                                 ; Find out final address

045752  201 01 0 00 040000               MOVEI   1,SELLAR                ; gset up to read LAR
045753  260 17 0 00 045737*              GO      LDCSR                   ; write CSR
045754  260 17 0 00 045413*              GO      RDLAR                   ; read LAR
045755  242 01 0 00 777777               LSH     1,-1                    ; position correctly
045756  202 01 0 00 046026               MOVEM   1,QADDRF                ; save it

                                 ; Check out channel logout status ...

045757  402 00 0 00 046033               SETZM   CSERR                   ; clear error status
045760  336 00 0 00 046040               SKIPN   CSTATF                  ; check logout status?
045761  254 00 0 00 045772               JRST    CEXWA4                  ; no - continue
045762  260 17 0 00 000000*              GO      CHDATA                  ; check logout data for errors
045763  332 00 0 00 000001               SKIPE   1                       ; any errors?
045764  474 15 0 00 000000               SETO    ERFLG,                  ; yes - set error flag
045765  202 01 0 00 046033               MOVEM   1,CSERR                 ; save error flags
045766  200 01 0 00 063304               MOVE    1,[CSTWRD,,CLOGBF]      ; move it from CSTWRD to
045767  251 01 0 00 046043               BLT     1,CLOGBF+2              ;   CLOGBF
045770  334 00 0 00 000000               SKIPA                           ; continue
```

```

                                 ; Error did occur

045771  474 15 0 00 000000       CEXWA3: SETO    ERFLG,                  ; set error flag
```

```
                                         ; Error did occur

045771  474 15 0 00 000000               CEXWA3: SETO    ERFLG,                    ; set error flag

                                         ; Exit

045772  332 00 0 00 064641'              CEXWA4: SKIPE   SDEBUG                    ; EXEC debug mode?
                                                 GO      [TMSGC  <CWAIT CSR >      ; yes - print debug
                                                          MOVE   CCSR              ;   data and continue
                                                          PNTHW
                                                          TMSG   < EBUF >
                                                          MOVE   CEBUFA
                                                          PNTOCS
                                                          TMSG   < ChnErr >
                                                          MOVE   CSERR
                                                          PNTOCS
045773  260 17 0 00 063312                                RTNJ]
045774  336 00 0 00 000015                       SKIPN   ERFLG                     ; error flag set?
045775  350 00 0 17 777775                       AOS     -3(P)                     ; no - set up good return
045776  350 00 0 17 777775                       AOS     -3(P)                     ; set up return
045777  350 00 0 17 777775                       AOS     -3(P)                     ; set up return
046000  262 17 0 00 000002                       RGET    (2,1,0)                   ; restore AC's
046003  263 17 0 00 000000                       RTN                               ; return

                                         ; Do data compare

046004  350 00 0 00 047012               CEXCOM: AOS     TSTSUB                    ; increment test segment
046005  210 01 0 00 046037                       MOVN    1,CTRAN                   ; get transfer type
046006  200 02 0 00 046036                       MOVE    2,CLEN                    ; get transfer length
046007  200 14 0 00 046035                       MOVE    PAT,CPAT                  ; get data pattern
046010  260 17 0 00 042620*                      GO      BUFCOM                    ; compare the data
046011  332 00 0 00 000001                       SKIPE   1                         ; any data compare errors?
046012  476 00 0 00 046034                       SETOM   CDERR                     ; yes - set data compare error flag
046013  332 00 0 00 000001                       SKIPE   1                         ; any data compare errors?
046014  474 15 0 00 000000                       SETO    ERFLG,                    ; yes - set error flag
046015  336 00 0 00 000015                       SKIPN   ERFLG                     ; error flag set?
046016  350 00 0 17 777775                       AOS     -3(P)                     ; no - set up good return
046017  350 00 0 17 777775                       AOS     -3(P)                     ; set up return
046020  350 00 0 17 777775                       AOS     -3(P)                     ; set up return
046021  262 17 0 00 000002                       RGET    (2,1,0)                   ; restore AC's
046024  263 17 0 00 000000                       RTN                               ; return
```

```
                                         ; Miscellaneous

046025  000000  000000                   QADDR:  0                                 ; initial starting address
                                                                                   ; final address
```

```
 9262
 9263                                           ; Miscellaneous
 9264
 9265   046025  000000  000000          QADDR:  0                                       ; initial starting address
 9266   046026  000000  000000          QADDRF: 0                                       ; final address
 9267   046027  000000  000000          CCSR:   0                                       ; final CSR data
 9268   046030  000000  000000          CTIME:  0                                       ; amount of time to let port run
 9269   046031  000000  000000          CEBUFC: 0                                       ; correct EBUF data
 9270   046032  000000  000000          CEBUFA: 0                                       ; actual EBUF data
 9271   046033  000000  000000          CSERR:  0                                       ; status error (0-no,xx-error flags)
 9272   046034  000000  000000          CDERR:  0                                       ; data compare error flag (0-no,-1-yes)
 9273   046035  000000  000000          CPAT:   0                                       ; data pattern used
 9274   046036  000000  000000          CLEN:   0                                       ; buffer length
 9275   046037  000000  000000          CTRAN:  0                                       ; transfer type (0-none,1-rd,2-wr
                                                                                        ;   3-rd/wr)
 9276
 9277   046040  000000  000000          CSTATF: 0                                       ; check logout status flag (0-n,-1-y)
 9278
 9279   046041                          CLOGBF: BLOCK   3                               ; logout area buffer
```

```
 9280
 9281                                   ;#*******************************************************************
 9282                                   ;* CCPNT - Print out error data
 9283                                   ;#*******************************************************************
```

```
                                                ;#*****************************************************************
                                                ;* CCPNT - Print out error data
                                                ;#*****************************************************************

046044  261 17 0 00 000000        CCPNT:  RPUT    (0,1)                   ; save AC's
046046  200 01 0 00 045442*               MOVE    1,SCOSW                 ; get switches

                                  ; Print start/end address

046047  603 01 0 00 000200                TLNE    1,TXTINH                ; full printout?
                                          GO      [TMSGC  <Start >        ; no
                                                   MOVE   QADDR
                                                   PNT4
                                                   TMSG   <  End >        ; print end address
                                                   MOVE   QADDRF
                                                   PNT4
046050  260 17 0 00 063330                         RTN]
046051  603 01 0 00 000200                TLNE    1,TXTINH                ; full printout
046052  254 00 0 00 046064                JRST    CCPNO                   ; no - continue
046053  037 00 0 00 062576                TMSGC   <Start Addr:  >         ; print start address
046054  200 00 0 00 046025                MOVE    QADDR
046055  037 04 0 00 000000                PNT4
046056  037 00 0 00 063337                TMSGC   <End Addr:    >         ; print end address
046057  200 00 0 00 046026                MOVE    QADDRF
046060  037 04 0 00 000000                PNT4
046061  037 00 0 00 063155                TMSGC   <Final CSR data:  >
046062  200 00 0 00 046027                MOVE    CCSR
046063  037 13 0 00 000000                PNTHW                           ; print it

                                  ; Check EBUF data

046064  200 00 0 00 046031        CCPNO:  MOVE    CEBUFC                  ; get correct EBUF data
046065  316 00 0 00 046032                CAMN    CEBUFA                  ; same as actual?
046066  254 00 0 00 046103                JRST    CCPN2                   ; yes - exit
046067  603 01 0 00 000200                TLNE    1,TXTINH                ; full printout?
046070  037 00 0 00 063005                TMSGC   <EBUF (C):  >           ; no
046071  607 01 0 00 000200                TLNN    1,TXTINH                ; full printout?
046072  037 00 0 00 062767                TMSGC   <EBUF - Correct:  >     ; yes
046073  200 00 0 00 046031                MOVE    CEBUFC
046074  037 13 0 00 000000                PNTHW
046075  603 01 0 00 000200                TLNE    1,TXTINH                ; full printout?
046076  037 00 0 00 063010                TMSGC   <     (A):  >           ; no
046077  607 01 0 00 000200                TLNN    1,TXTINH                ; full printout?
046100  037 00 0 00 062773                TMSGC   <         Actual:  >    ; yes
046101  200 00 0 00 046032                MOVE    CEBUFA
046102  037 13 0 00 000000                PNTHW
```

```
                                  ; Print status errors

046103  200 01 0 00 046046*       CCPN2:  MOVE    1,SCOSW                 ; get switches
                                          TLNE    1,TXTINH                ; full printout?
```

```
                                        ; Print status errors

046103  200 01 0 00 046046*     CCPN2:  MOVE    1,SCOSW                 ; get switches
046104  603 01 0 00 000200              TLNE    1,TXTINH                ; full printout?
046105  254 00 0 00 046111              JRST    CCPN3                   ; no - continue
046106  201 01 0 00 046041              MOVEI   1,CLOGBF                ; get address of logout area
046107  260 17 0 00 042605*             GO      LOGPNT                  ; print it out
046110  254 00 0 00 046135              JRST    CCPN4                   ; continue

046111  332 01 0 00 046033      CCPN3:  SKIPE   1,CSERR                 ; get status bits
046112  037 00 0 00 063343              TMSGC   <Status errors:  >
046113  603 01 0 00 200000              TLNE    1,(MPER)                ; mem parity error
046114  037 00 0 00 063347              TMSG    <MPERR >
046115  603 01 0 00 100000              TLNE    1,(ADRPE)               ; address parity error
046116  037 00 0 00 063351              TMSG    <ADRPE >
046117  603 01 0 00 040000              TLNE    1,(NOTWCO)              ; not word count zero
046120  037 00 0 00 063353              TMSG    <NOTWCO >
046121  603 01 0 00 020000              TLNE    1,(NEXM)                ; non-existant memory
046122  037 00 0 00 063355              TMSG    <NEXM >
046123  603 01 0 00 000400              TLNE    1,(LAXER)               ; last xfer error
046124  037 00 0 00 063357              TMSG    <LAXER >
046125  603 01 0 00 000200              TLNE    1,(IPAER)               ; Port error
046126  037 00 0 00 063361              TMSG    <IPAER >
046127  603 01 0 00 000100              TLNE    1,(LGWC)                ; long word count
046130  037 00 0 00 063363              TMSG    <LGWC >
046131  603 01 0 00 000040              TLNE    1,(SHWC)                ; short word count
046132  037 00 0 00 063365              TMSG    <SHWC >
046133  603 01 0 00 000020              TLNE    1,(OVN)                 ; overrrun
046134  037 00 0 00 063367              TMSG    <OVN >

                                        ; Print data compare errors

046135  336 00 0 00 046034      CCPN4:  SKIPN   CDERR                   ; any data compare errors?
046136  254 00 0 00 046143              JRST    CCPNX                   ; no - exit
046137  200 01 0 00 046037              MOVE    1,CTRAN                 ; get transfer type
046140  200 02 0 00 046036              MOVE    2,CLEN                  ; get transfer length
046141  200 14 0 00 046035              MOVE    PAT,CPAT                ; get data pattern
046142  260 17 0 00 046010*             GO      BUFCOM                  ; compare the data

                                        ; Exit

046143  262 17 0 00 000001      CCPNX:  RGET    (1,0)                   ; restore AC's
046145  263 17 0 00 000000              RTN                             ; return
```

```
                                ;#*****************************************************************
                                ;* IEXEC - Interrupt test execute routine
                                ;
                                ; This routine is used to execute a interrupt type test.   It  allows
```

```
;*************************************************************************
;* IEXEC - Interrupt test execute routine
;
; This routine is used to execute a interrupt type test.  It  allows
; the interrupt system to be set up and interrupts to be fielded.
;
; This  routine  is  given  a  pointer  to an 'Interrupt Test Execute
; Table'.  It  then  executes  one  entry  and returns +1,+2,+3 or +4
; depending on the type of entry and the result.
;
; Argument:     AC6 - Address of 'Interrupt Test Execute Table'
;
; Return:       +1 - End of table was seen.
;               +2 - A routine was called (just do next entry).
;               +3 - Error return.
;               +4 - Success return.
;
;               IADDR - Initial starting address
;               IADDRF - Final address
;               ICSR - Final CSR data
;               ITIME - Amount of time to let port run
;               IFLAG - Interrupt should occur data
;                 bits 0-2 - IOP function
;                            0 - non-vectored
;                            1 - vectored
;                            2 - examine
;                            3 - deposit
;                            4 - increment
;                            5 - decrement
;                            6 - examine/increment
;                 bits 3-8 - type of interrupt
;                            0 - vectored
;                            1 - non-vectored on PI level 1
;                            .         ...         ...
;                            7                          7
;                           77 - no interrupt should occur
;                 bits 9-35 - address to execute (for vectored
;                             or non-vectored interrupts) or the
;                             location to increment, decrement,
;                             examine, or deposit
;               IITYPE - Type of interrupt which occurred
;                           -1 - none
;                            0 - vectored
;                            1 - non-vectored on PI level 1
;                            .         ...   ...
;                            7 - non-vectored on PI level 7
```

```
;               IINTPC - Interrupt PC
;               IIADR - Address of location to be inc/dec/exa/dep
;               IIINCC - Contents of location to be inc/dec (correct)
;               IIINCA - Contents of location to be inc/dec (actual)
;                       IINTPC - Interrupt PC
;                       IIADR - Address of location to be inc/dec/exa/dep
;                       IIINCC - Contents of location to be inc/dec (correct)
;                       IIINCA - Contents of location to be inc/dec (actual)
;                       IIEXDC - Contents of location exam/depos (correct)
;                       IIEXDA - Contents of location exam/depos (actual)
;                       IEBUF - EBUF contents (of 2901 register 5)
;
;                       AC6 - Updated to point to the next table location
;
; Interrupt Test Execute Table Format:
;
;                       Entry
;                        ...
;                       ILAST          ; last entry (zero)
;
; 'Entry' is of the form:
;
; Bits 0-3 - Type of entry
;    0 - Done with table    (ILAST)       6 - Call routine/exit  (ICALLC)
;    1 - Set up examine     (ISETEX)      7 - Wait for interrupt (IWAIT)
;    2 - Set up deposit     (ISETDE)     10 - Select new tbl addr(IJRST)
;    3 - Set up inc/dec     (ISETID)     11 - Start port         (ISTART)
;    4 - Set up v/nv int    (ISETIN)     12 - Exit test (if ERFLG set) (IEXIT)
;    5 - Call routine       (ICALL)
;
; Bits 4-35 - Defined per entry type.
;
;     Entry type 0 (ILAST) -  All zero - Indicates end of table
;
;     Entry type 1 (ISETEX) - Bits 4-17 - IOP function
;                                 18-35 - Address of list containing addr to
;                                         be examined and the data to use
;
;     Entry type 2 (ISETDE) - Bits 4-17 - IOP function
;                                 18-35 - Address of list containing addr to
;                                         be deposited and the data to use
;
;     Entry type 3 (ISETID) - Bits 4-17 - IOP function
;                                 18-35 - Address of list containing addr to
;                                         be inc/dec and data to use
;
;     Entry type 4 (ISETIN) - Bits 4-17 - IOP function
;                                 18-35 - Interrupt type expected (0-vectored,
;                                         n-nonvectored on PI level n)
;
;     Entry type 5 (ISETE1) - Bits 4-17 - IOP function
;                                 18-35 - Address of list containing addr to
;                                         be examined and the data to use
;
```

```

;     Entry type 6 (ICALL) -  Bits 4-35 - Address of routine to call
;
;     Entry type 7 (ICALLC) - Bits 4-35 - Address of routine to call
```

```
                                   ;    Entry type 6 (ICALL) -  Bits 4-35 - Address of routine to call
                                   ;
                                   ;    Entry type 7 (ICALLC) - Bits 4-35 - Address of routine to call
                                   ;
                                   ;    Entry type 10 (IWAIT) - Bits 4-35 - Time to wait (in milliseconds)
                                   ;       After the wait is complete, and interrupt data gotten, the PI
                                   ;       system is shut off and R5 is read by restarting port at location
                                   ;       1100, then stopping the port and reading the EBUF.
                                   ;
                                   ;    Entry type 11 (IJRST) -  Bits 4-35 - Table address to continue
                                   ;
                                   ;    Entry type 12 (ISTART) - Bits 4-35 - CRAM address to start at
                                   ;
                                   ;    Entry type 13 (IEXIT) - Bits 4-35 - Unused
                                   ;#**********************************************************************

046146  261 17 0 00 000000         IEXEC:  RPUT    (0,1,2)                 ; save some AC's

                                   ; Check if end of table

046151  200 02 0 06 000000                 MOVE    2,(6)                   ; get entry
046152  350 00 0 00 000006                 AOS     6                       ; point to next entry
046153  322 02 0 00 046157                 JUMPE   2,IEXEX1                ; end of table - yes - exit

                                   ; Determine type of entry and dispatch

046154  135 01 0 00 063013                 LDB     1,[POINT 4,2,3]         ; get entry
                                           XCT     [JRST   IEXERR          ; dispatch on it
                                                     JRST   IEXSEX
                                                     JRST   IEXSDE
                                                     JRST   IEXSID
                                                     JRST   IEXSIN
                                                     JRST   IEXSEI
                                                     JRST   IEXCAL
                                                     JRST   IEXCAC
                                                     JRST   IEXWAI
                                                     JRST   IEXJRS
                                                     JRST   IEXSTR
                                                     JRST   IEXEXI
                                                     JRST   IEXERR
046155  256 00 0 01 063370                           JRST   IEXERR](1)

                                   ; Error in Test Execute Table

046156  037 00 0 00 063176         IEXERR: TMSGCD  <Data Test Execute Dispatch Table Error - Entry out of range>
046157  332 00 0 00 064641'        IEXEX1: SKIPE   SDEBUG                  ; EXEC debug mode?
046160  037 00 0 00 030242                 PCRL                            ; yes - a final CRLF
046161  262 17 0 00 000002                 RGET    (2,1,0)                 ; restore AC's
046164  263 17 0 00 000000                 RTN                             ; return
```

```
                                   ; Set up an examine transfer

046165  135 01 0 00 063406         IEXSEX: LDB     1,[POINT 18,2,35]       ; get address of list
```

```
 9518                                          ; Set up an examine transfer
 9519
 9520
 9521  046165  135 01 0 00 063406              IEXSEX: LDB     1,[POINT 18,2,35]       ; get address of list
 9522  046166  200 00 0 01 000000                      MOVE    (1)                     ; get address
 9523  046167  202 00 0 00 046600                      MOVEM   IIADR                   ; save it
 9524  046170  200 00 0 01 000001                      MOVE    1(1)                    ; get data
 9525  046171  202 00 0 00 046601                      MOVEM   IIINCC                  ; save it
 9526  046172  202 00 0 00 046602                      MOVEM   IIINCA                  ; save it
 9527  046173  202 00 0 00 046605                      MOVEM   IIDEPC                  ; save it
 9528  046174  202 00 0 00 046606                      MOVEM   IIDEPA                  ; save it
 9529  046175  202 00 0 00 046603                      MOVEM   IIEXAC                  ; save it
 9530  046176  200 00 0 00 063213                      MOVE    [400000,,1]             ; get null data
 9531  046177  202 00 0 00 046604                      MOVEM   IIEXAA                  ; save it as actual
 9532  046200  200 01 0 00 046600                      MOVE    1,IIADR                 ; include address
 9533  046201  201 00 0 00 000002                      MOVEI   2                       ; get IOP function
 9534  046202  137 00 0 00 063407                      DPB     [POINT 3,1,2]           ; save it
 9535  046203  201 00 0 00 000077                      MOVEI   77                      ; get 'no interrupt' code
 9536  046204  137 00 0 00 063410                      DPB     [POINT 6,1,8]           ; save it
 9537  046205  202 01 0 00 046574                      MOVEM   1,IFLAG                 ; save interrupt type
 9538  046206  332 00 0 00 064641'                     SKIPE   SDEBUG                  ; test debug mode?
 9539                                                  GO      [TMSGC  <ISETEX  Iflag >; yes - print debug
 9540                                                            MOVE   IFLAG           ;   data and continue
 9541                                                            PNTHW
 9542                                                            TMSG   < Adr >
 9543                                                            MOVE   1,IIADR
 9544                                                            PNTOCS
 9545                                                            TMSG   < Data >
 9546                                                            MOVE   1,IIEXAC
 9547                                                            PNTOCS
 9548  046207  261 17 0 00 063421                                RTN]
 9549  046210  350 00 0 17 777775                      AOS     -3(P)                   ; set up proper return
 9550  046211  262 17 0 00 000002                      RGET    (2,1,0)                 ; restore AC's
 9551  046214  263 17 0 00 000000                      RTN                             ; return
 9552
 9553                                          ; Set up a deposit transfer
 9554
 9555  046215  135 01 0 00 063406              IEXSDE: LDB     1,[POINT 18,2,35]       ; get address of list
 9556  046216  200 00 0 01 000000                      MOVE    (1)                     ; get address
 9557  046217  202 00 0 00 046600                      MOVEM   IIADR                   ; save it
 9558  046220  200 00 0 01 000001                      MOVE    1(1)                    ; get data
 9559  046221  202 00 0 00 046601                      MOVEM   IIINCC                  ; save it
 9560  046222  202 00 0 00 046602                      MOVEM   IIINCA                  ; save it
 9561  046223  202 00 0 00 046603                      MOVEM   IIEXAC                  ; save it
 9562  046224  202 00 0 00 046604                      MOVEM   IIEXAA                  ; save it
 9563  046225  202 00 0 00 046605                      MOVEM   IIDEPC                  ; save it
 9564  046226  200 00 0 00 063213                      MOVE    [400000,,1]             ; get null data
 9565  046227  202 00 0 00 046606                      MOVEM   IIDEPA                  ; save it as actual
 9566  046230  200 01 0 00 046600                      MOVE    1,IIADR                 ; include address
 9567  046231  201 00 0 00 000003                      MOVEI   3                       ; get IOP function
 9568  046232  137 00 0 00 063407                      DPB     [POINT 3,1,2]           ; save it
```

```
 9569  046233  201 00 0 00 000077                      MOVEI   77                      ; get 'no interrupt' code
 9570  046234  137 00 0 00 063410                      DPB     [POINT 6,1,8]           ; save it
 9571  046235  202 01 0 00 046574                      MOVEM   1,IFLAG                 ; save interrupt type
 9572  046236  332 00 0 00 064641'                     SKIPE   SDEBUG                  ; test debug mode?
```

```
046233  201 00 0 00 000077                   MOVEI   77                        ; get 'no interrupt' code
046234  137 00 0 00 063410                   DPB     [POINT 6,1,8]             ; save it
046235  202 01 0 00 046574                   MOVEM   1,IFLAG                   ; save interrupt type
046236  332 00 0 00 064641'                  SKIPE   $DEBUG                    ; test debug mode?
                                             GO      [TMSGC  <ISETDE  Iflag >; yes - print debug
                                                      MOVE   IFLAG             ;   data and continue
                                                      PNTHW
                                                      TMSG   < Adr >
                                                      MOVE   1,IIADR
                                                      PNTOCS
                                                      TMSG   < Data >
                                                      MOVE   1,IIDEPC
                                                      PNTOCS
046237  260 17 0 00 063437                            RTN]
046240  350 00 0 17 777775                   AOS     -3(P)                     ; set up proper return
046241  262 17 0 00 000002                   RGET    (2,1,0)                   ; restore AC's
046244  263 17 0 00 000000                   RTN                               ; return

                                     ; Set up an inc/dec operation

046245  135 01 0 00 063406           IEXSID: LDB     1,[POINT 18,2,35]         ; get address of list
046246  200 00 0 01 000000                   MOVE    (1)                       ; get address
046247  202 00 0 00 046600                   MOVEM   IIADR                     ; save it
046250  200 00 0 01 000001                   MOVE    1(1)                      ; get data
046251  202 00 0 00 046603                   MOVEM   IIEXAC                    ; save it
046252  202 00 0 00 046604                   MOVEM   IIEXAA                    ; save it
046253  202 00 0 00 046605                   MOVEM   IIDEPC                    ; save it
046254  202 00 0 00 046606                   MOVEM   IIDEPA                    ; save it
046255  202 00 0 00 046601                   MOVEM   IIINCC                    ; save it
046256  200 00 0 00 063213                   MOVE    [400000,,1]               ; get null data
046257  202 00 0 00 046602                   MOVEM   IIINCA                    ; save it as actual
046260  200 01 0 00 046600                   MOVE    1,IIADR                   ; include address
046261  201 00 0 00 000004                   MOVEI   4                         ; get IOP function
046262  137 00 0 00 063407                   DPB     [POINT 3,1,2]             ; save it
046263  201 00 0 00 000077                   MOVEI   77                        ; get 'no interrupt' code
046264  137 00 0 00 063410                   DPB     [POINT 6,1,8]             ; save it
046265  202 01 0 00 046574                   MOVEM   1,IFLAG                   ; save interrupt type
046266  332 00 0 00 064641'                  SKIPE   $DEBUG                    ; test debug mode?
                                             GO      [TMSGC  <ISETID  Iflag >; yes - print debug
                                                      MOVE   IFLAG             ;   data and continue
                                                      PNTHW
                                                      TMSG   < Adr >
                                                      MOVE   1,IIADR
                                                      PNTOCS
                                                      TMSG   < Data >
                                                      MOVE   1,IIINCC
                                                      PNTOCS
046267  260 17 0 00 063455                            RTN]
046270  350 00 0 17 777775                   AOS     -3(P)                     ; set up proper return
046271  262 17 0 00 000002                   RGET    (2,1,0)                   ; restore AC's
046274  263 17 0 00 000000                   RTN                               ; return
```

```
                                     ; Set up an examine/increment operation

046275  135 01 0 00 063406           IEXSEI: LDB     1,[POINT 18,2,35]         ; get address of list
```

```


                                        ; Set up an examine/increment operation

046275  135 01 0 00 063406      IEXSEI: LDB     1,[POINT 18,2,35]       ; get address of list
046276  200 00 0 01 000000              MOVE    (1)                     ; get address
046277  202 00 0 00 046600              MOVEM   IIADR                   ; save it
046300  200 00 0 01 000001              MOVE    1(1)                    ; get data
046301  350 00 0 00 000000              AOS                             ; get correct data
046302  202 00 0 00 046603              MOVEM   IIEXAC                  ; save it
046303  202 00 0 00 046605              MOVEM   IIDEPC                  ; save it
046304  202 00 0 00 046606              MOVEM   IIDEPA                  ; save it
046305  202 00 0 00 046601              MOVEM   IIINCC                  ; save it
046306  200 00 0 00 063213              MOVE    [400000,,1]             ; get null data
046307  202 00 0 00 046602              MOVEM   IIINCA                  ; save it as actual
046310  202 00 0 00 046604              MOVEM   IIEXAA                  ; save it as actual
046311  200 01 0 00 046600              MOVE    1,IIADR                 ; include address
046312  201 00 0 00 000006              MOVEI   6                       ; get IOP function
046313  137 00 0 00 063407              DPB     [POINT 3,1,2]           ; save it
046314  201 00 0 00 000077              MOVEI   77                      ; get 'no interrupt' code
046315  137 00 0 00 063410              DPB     [POINT 6,1,8]           ; save it
046316  202 01 0 00 046574              MOVEM   1,IFLAG                 ; save interrupt type
046317  332 00 0 00 064641'             SKIPE   SDEBUG                  ; test debug mode?
                                        GO      [TMSGC  <ISETEI  Iflag >; yes - print debug
                                                  MOVE    IFLAG         ;   data and continue
                                                  PNTHW
                                                  TMSG    < Adr >
                                                  MOVE    1,IIADR
                                                  PNTOCS
                                                  TMSG    < Data >
                                                  MOVE    1,IIEXAC
                                                  PNTOCS
                                                  RTN]
046320  260 17 0 00 063473              AOS     -3(P)                   ; set up proper return
046321  350 00 0 17 777775              RGET    (2,1,0)                 ; restore AC's
046322  262 17 0 00 000002              RTN                             ; return
046325  263 17 0 00 000000

                                        ; Set up interrupt (non-vectored or vectored)

046326  260 17 0 00 042714*     IEXSIN: GO      INITPI                  ; init PI system
046327  260 17 0 00 042715*             GO      SETVEC                  ; set up vector addresses
046330  200 01 0 00 046570              MOVE    1,IADDR                 ; get address
046331  135 00 0 00 063105              LDB     [POINT 14,2,17]         ; get IOP function
046332  137 00 0 00 063407              DPB     [POINT 3,1,2]           ; position correctly for IFLAG
046333  135 00 0 00 063406              LDB     [POINT 18,2,35]         ; get interrupt type expected
046334  137 00 0 00 063410              DPB     [POINT 6,1,8]           ; position correctly for IFLAG
046335  202 01 0 00 046574              MOVEM   1,IFLAG                 ; save interrupt type
046336  200 00 0 00 046601              MOVE    IIINCC                  ; get correct increment data
046337  202 00 0 00 046602              MOVEM   IIINCA                  ; save actual increment data
046340  200 00 0 00 046603              MOVE    IIEXAC                  ; get correct examine data
```

```


                                        ; Check CI port

032470  200 01 0 00 047004              TSTSGO: MOVE    1,CISEL                 ; get CI select word
032471  606 01 0 00 000002                      TRNN    1,2                     ; port selectable?
032472  254 00 0 00 032504                      JRST    TSTSEX                  ; no - exit
032473  200 00 0 00 054462                      MOVE    [FMSGC  <Test CI port?>]
032474  260 17 0 00 032464*                     GO      TTYYES                  ; check for Y or N
032475  624 01 0 00 000004                      TRZA    1,4                     ; no - set no
032476  660 01 0 00 000004                      TRO     1,4                     ; yes - set yes
032477  202 01 0 00 047004                      MOVEM   1,CISEL                 ; save selection

                                        ; Set up MBCN properly

032500  200 16 0 00 047001                      MOVE    MBCN,PORTNI             ; get NI device code
032501  200 01 0 00 047003                      MOVE    1,NISEL                 ; get NI select word
032502  606 01 0 00 000004                      TRNN    1,4                     ; selected?
032503  200 16 0 00 047002                      MOVE    MBCN,PORTCI             ; no - get CI device code

                                        ; Print selection

032504  200 00 0 00 047003              TSTSEX: MOVE    NISEL                   ; get NI select word
032505  602 00 0 00 000004                      TRNE    4                       ; selected?
032506  037 00 0 00 054463                      TMSGC   <[NI port selected]>    ; yes - print such
032507  200 00 0 00 047004                      MOVE    CISEL                   ; get CI select word
032510  602 00 0 00 000004                      TRNE    4                       ; selected?
032511  037 00 0 00 054470                      TMSGC   <[CI port selected]>    ; yes - print such
032512  200 00 0 00 047003                      [illegible] NISEL               ; get NI select word
032513  434 00 0 00 047004                      IOR     CISEL                   ; include CI select word
032514  606 00 0 00 000004                      TRNN    [illegible]             ; any selected?
032515  037 00 0 00 054475                      TMSGC   [illegible]orts selected]>   ; no - print such
032516  037 00 0 00 030242                      PCRL
032517  263 17 0 00 000000                      RTN                             ; return
```

```

                                                ;#*****************************************************************
                                                ; PTEST - Determine port under test and set up MBCN
                                                ;  TESTP - Print port under test (if not already known)
                                                ;******************************************************************

032520  261 17 0 00 000000      PTEST:  PUT     0                               ; save AC0
032521  316 16 0 00 047001              CAMN    MBCN,PORTNI                     ; NI port already selected?
032522  254 00 0 00 032535              JRST    PTEST0                          ; yes - continue
032523  316 16 0 00 047002              CAMN    MBCN,PORTCI                     ; no - CI port already selected?
032524  254 00 0 00 032535              JRST    PTEST0                          ; yes - continue
032525  200 16 0 00 047001              MOVE    MBCN,PORTNI                     ; no - set to NI port
032526  200 00 0 00 047003              MOVE    NISEL                           ; get NI select bits
032527  602 00 0 00 000004              TRNE    4                               ; select possible?
032530  254 00 0 00 032535              JRST    PTEST0                          ; yes - continue
032531  200 16 0 00 047002              MOVE    MBCN,PORTCI                     ; no - set to CI port
032532  200 00 0 00 047004              MOVE    CISEL                           ; get CI select bits
032533  606 00 0 00 000004              TRNN    4                               ; select possible?
032534  400 16 0 00 000000              SETZ    MBCN,                           ; no - indicate such
032535  262 17 0 00 000000      PTEST0: GET     0                               ; restore AC0
032536  263 17 0 00 000000              RTN                                     ; return

032537  261 17 0 00 000000      PTESTP: PUT     0                               ; save AC0
032540  200 00 0 00 047003              MOVE    NISEL                           ; get NI select word
032541  404 00 0 00 047004              AND     CISEL                           ;   and CI select word
032542  606 00 0 00 000004              TRNN    4                               ; both selected?
032543  254 00 0 00 032550              JRST    PTESTX                          ; no - continue
032544  316 16 0 00 047001              CAMN    MBCN,PORTNI                     ; NI port already selected?
032545  037 00 0 00 054502              TMSGCD  <[Port under test - NI]>; yes - print such
032546  316 16 0 00 047002              CAMN    MBCN,PORTCI                     ; CI port already selected?
032547  037 00 0 00 054510              TMSGCD  <[Port under test - CI]>; yes - print such
032550  336 00 0 00 000016      PTESTX: SKIPN   MBCN                            ; any port selected?
032551  037 00 0 00 054516              TMSGCD  <[Port under test - none]> ; no - print such
032552  262 17 0 00 000000              GET     0                               ; restore AC0
032553  263 17 0 00 000000              RTN                                     ; exit
```

```

                                        ; Initialze test class/number at beginning of pass

033200  200 00 0 00 064634'                     MOVE    SAVADD                  ; get test dispatch address
033201  202 00 0 00 047015                      MOVEM   TSTADD                  ; save it
033202  260 17 0 00 033155*                     GO      SWITT                   ; get the pgm switches
033203  201 01 0 00 000001                      MOVEI   1,1                     ; initialize repeat count
033204  603 00 0 00 000400                      TLNE    RELIAB                  ; reliability switch set?
033205  201 01 0 00 000004                      MOVEI   1,4                     ; yes - set up count of 4
033206  202 01 0 00 047027                      MOVEM   1,TSTREL                ; save it

                                        ; Now actually do the tests

033207  200 00 0 00 047027              TSTD8:  MOVE    TSTREL                  ; get reliability count
033210  202 00 0 00 047030                      MOVEM   TSTCNT                  ; save it
033211  260 17 0 00 043023                      GO      SELTST                  ; select next test
033212  254 00 0 00 033130                        JRST  TSTD5                   ; end of tests
033213  254 00 0 00 033255                        JRST  TSTD12                  ; not a valid test
033214  200 16 0 00 047006              TSTD9:  MOVE    MBCN,UUT                ; valid test - set up MBCN
033215  260 17 1 00 047013                      GO      @TSTPC                  ; dispatch to test
033216  254 00 0 00 033255                      JRST    TSTD12                  ; disabled test - skip it
033217  332 00 0 00 047023                      SKIPE   TSTINH                  ; inhibit fault isolation?
033220  254 00 0 00 033246                      JRST    TSTD11                  ; yes - continue
033221  332 00 0 00 064606'                     SKIPE   IDEBUG                  ; fault isolation debug mode?
033222  260 17 0 00 034267                      GO      FLTCHK                  ; yes - check if this test should fail
033223  322 15 0 00 033246                      JUMPE   ERFLG,TSTD11            ; test did not fail - continue
```

```


                                      ; If this is the first failure, also set up new test dispatch table (as
                                      ; specified in the test itself).

033224  332 00 0 00 064601'                   SKIPE   ERRCNT                    ; any errors yet?
033225  254 00 0 00 033231                    JRST    TSTD10                    ; no - continue
033226  200 00 0 00 047013                    MOVE    TSTPC                     ; get test PC
033227  271 00 0 00 000003                    ADDI    3                         ; point to new table address
033230  202 00 0 00 047015                    MOVEM   TSTADD                    ; set up new dispatch address

                                      ; Handle fault isolation

033231  350 00 0 00 064601'           TSTD10: AOS     ERRCNT                    ; increment error count
033232  200 03 0 00 047024                    MOVE    3,TSTHIG                  ; get addr of high prob module list
033233  201 04 0 00 000014                    MOVEI   4,^D12                    ; weight of a high prob module
033234  332 00 0 00 064606'                   SKIPE   IDEBUG                    ; fault isolation debug mode?
                                              GO      [TMSGC  <*** >            ; yes - print header ...
                                                        MOVE   1,TSTCLS         ; get test class
                                                        MOVE   INDLIS(1)        ; get class description
                                                        PNTSIX                  ; print it
                                                        MOVE   TSTNUM           ; get test number
                                                        PNTOCS                  ; print it
                                                        TMSG   < ***>
                                                        TMSGC  <High networks:  >
                                                        RTN]
033235  260 17 0 00 055355
033236  260 17 0 00 034251                    GO      FLTHAN                    ; go to fault handler
033237  200 03 0 00 047025                    MOVE    3,TSTLOW                  ; get addr of low prob module list
033240  201 04 0 00 000006                    MOVEI   4,^D6                     ; weight of a low prob module
033241  332 00 0 00 064606'                   SKIPE   IDEBUG                    ; fault isolation debug mode?
033242  037 00 0 00 055366                    TMSGC   <Low networks:  >         ; yes - print header
033243  260 17 0 00 034251                    GO      FLTHAN                    ; go to fault handler
033244  332 00 0 00 064606'                   SKIPE   IDEBUG                    ; fault isolation debug mode?
033245  037 00 0 00 030242                    PCRL                              ; yes - end with a CRLF
```

```

                                ; Local Storage Tests

034132  000000000000#                   TSTU34          ; MP34 - Basic Local Store Test
034133  000000000000#                   TSTU35          ; MP35 - Local Store Data Test
034134  000000000000#                   TSTU36          ; MP36 - Local Store Address Test
034135  000000000000#                   TSTU37          ; MP37 - Local Store RAM Mode Test

                                ; Miscellaneous

034136  000000000000#                   TSTU40          ; MP40 - Jump MUX Test

                                ; Interrupt Logic

034137  000000000000#                   TSTE60          ; EB60 - PI Level n/0 Interrupts
034140  000000000000#                   TSTE61          ; EB61 - Check PIA Level 0
034141  000000000000#                   TSTE62          ; EB62 - Check Interrupt on Level 1-7
034142  000000000000#                   TSTE63          ; EB63 - CSR 05 'Rqst Interrupt' Clearing
034143  000000000000#                   TSTE64          ; EB64 - IOP Function 0 - 40+2N Interrupt
034144  000000000000#                   TSTE65          ; EB65 - IOP Function 1 - 40+2N Interrupt
034145  000000000000#                   TSTE70          ; EB70 - IOP Function 4 - Examine
034146  000000000000#                   TSTE71          ; EB71 - IOP Function 5 - Deposit
034147  000000000000#                   TSTE72          ; EB72 - IOP Function 7 - Examine/Increment

                                ; Basic CBUS Tests

034150  000000000000#                   TSTC1           ; CB1 - Fmtr Data Loopback
034151  000000000000#                   TSTC2           ; CB2 - PLI Buffer Data Loopback
034152  000000000000#                   TSTC11          ; CB11 - Fmtr Cleared by 'Port Clear'

                                ; Condition Code Tests

034153  000000000000#                   TSTU14          ; MP14 - CCCbusAvail Test
034154  000000000000#                   TSTU15          ; MP15 - CCGrntCSR Test
034155  000000000000#                   TSTU16          ; MP16 - CCFEQ0 Test
034156  000000000000#                   TSTU17          ; MP17 - CCCSRChng Test
034157  000000000000#                   TSTU20          ; MP20 - CCEbParErr Test
034160  000000000000#                   TSTU21          ; MP21 - CCRcvrBufAFul Test
034161  000000000000#                   TSTU22          ; MP22 - CCRcvrBufBFul Test
034162  000000000000#                   TSTU23          ; MP23 - CCXmtrAttn Test
034163  000000000000#                   TSTU24          ; MP24 - CCEbusRqst Test
034164  000000000000#                   TSTU25          ; MP25 - CCIntrActive Test
034165  000000000000#                   TSTU26          ; MP26 - CCMBSign Test
034166  000000000000#                   TSTU27          ; MP27 - CCMVParChk Test
034167  000000000000#                   TSTU30          ; MP30 - CCCbusParErr Test
034170  000000000000#                   TSTU31          ; MP31 - CCPliParErr Test
034171  000000000000#                   TSTU32          ; MP32 - CCChanErr Test
034172  000000000000#                   TSTU33          ; MP33 - CCCbLstWd Test

                                ; CBUS Data Transfer

034173  000000000000#                   TSTC3           ; CB3 - CBUS to EBUF Data Transfer
```

```
034174  000000000000#               TSTC4           ; CB4 - 2901 to CBUS Data Transfer
034175  000000000000#               TSTC5           ; CB5 - CBUS to 2901 Multiple Word Transfer
034176  000000000000#               TSTC6           ; CB6 - 2901 to CBUS Multiple Word Transfer
034177  000000000000#               TSTC7           ; CB7 - CBUS to EBUS Data Transfer
034200  000000000000#               TSTC10          ; CB10 - EBUS to CBUS Data Transfer

                            ; Formatter Tests

034201  000000000000#               TSTC12          ; CB12 - Fmtr Basic Up Shifting Test
034202  000000000000#               TSTC13          ; CB13 - Fmtr Up Shift 4 Test - BUS Ctl=Cbus
034203  000000000000#               TSTC14          ; CB14 - Fmtr Up Shift 4 Test - BUS Ctl=Fmtr
034204  000000000000#               TSTC15          ; CB15 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=0
034205  000000000000#               TSTC16          ; CB16 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=0
034206  000000000000#               TSTC17          ; CB17 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=1
034207  000000000000#               TSTC20          ; CB20 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=1
034210  000000000000#               TSTC21          ; CB21 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=0
034211  000000000000#               TSTC22          ; CB22 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=0
034212  000000000000#               TSTC23          ; CB23 - Fmtr Up Shift 8 (from PLIN) Test - SELRHTNIB=1
034213  000000000000#               TSTC24          ; CB24 - Fmtr Up Shift 4 (from PLIN) Test - SELRHTNIB=1
034214  000000000000#               TSTC25          ; CB25 - Fmtr Basic Down Shifting Test
034215  000000000000#               TSTC26          ; CB26 - Fmtr Down Shift 4 Test
034216  000000000000#               TSTC27          ; CB27 - Fmtr Down Shift 4 (from PLIN) Test SELRHTNIB=0
034217  000000000000#               TSTC30          ; CB30 - Fmtr Down Shift 4 (from PLIN) Test SELRHTNIB=0
034220  000000000000#               TSTC31          ; CB31 - Fmtr Down Shift 4 (from PLIN) Test SELRHTNIB=1
034221  000000000000#               TSTC32          ; CB32 - Fmtr Down Shift 4 (from PLIN) Test SELRHTNIB=1
034222  000000000000#               TSTC33          ; CB33 - Fmtr Down Shift 8 (from PLIN) Test SELRHTNIB=0
034223  000000000000#               TSTC34          ; CB34 - Fmtr Down Shift 8 (from PLIN) Test SELRHTNIB=0
034224  000000000000#               TSTC35          ; CB35 - Fmtr Down Shift 8 (from PLIN) Test SELRHTNIB=1
034225  000000000000#               TSTC36          ; CB36 - Fmtr Down Shift 4 (from PLIN) Test SELRHTNIB=1
034226  000000000000#               TSTC37          ; CB37 - Fmtr Ring Buffer Test

                            ; Miscellaneous CBUS module tests

034227  000000000000#               TSTC40          ; CB40 - CMUX Selected by PLINTOCMUX
034230  000000000000#               TSTC41          ; CB41 - CBUF Loaded Correctly
034231  000000000000#               TSTC42          ; CB42 - PMUX Selected by MPCBUFTOPLOUT
034232  000000000000#               TSTC43          ; CB43 - PMUX Selected by MPCBUFTOPLOUT/MPZEROLFTNIB
034233  000000000000#               TSTC44          ; CB44 - DMUX Selected Correctly
034234  000000000000#               TSTC45          ; CB45 - CBUS Parity Checking - CBUS to CBUS Module
034235  000000000000#               TSTC46          ; CB46 - CBUS Parity Checking - CBUS Module to CBUS
034236  000000000000#               TSTC47          ; CB47 - CBUS Parity Checking - CBUS to CBUS Module
034237  000000000000#               TSTC50          ; CB50 - T Field Timing Test
034240  000000000000#               TSTC51          ; CB51 - CBUS Store
034241  000000000000#               TSTC52          ; CB52 - CBUS Input Buffer
034242  000000000000#               TSTC53          ; CB53 - Parity Predictor - CBIN to Fmtr/MPINDSTCOMP=0
034243  000000000000#               TSTC54          ; CB54 - Parity Predictor - CBIN to Fmtr/MPINDSTCOMP=1
034244  000000000000#               TSTC55          ; CB55 - Parity Predictor - Fmtr to PLOUT
034245  000000000000#               TSTC56          ; CB56 - Parity Predictor - PLIN to Fmtr/MPCOREDUMP=0
034246  000000000000#               TSTC57          ; CB57 - Parity Predictor - PLIN to Fmtr/MPCOREDUMP=1
034247  000000000000#               TSTC60          ; CB60 - Parity Predictor - Fmtr to CBOUT
034250  777777  777777              -1              ; end of tests
```

```

                                 ;#*********************************************************************
                                 ;* .DBCMD - List of acceptable commands
                                 ;#*********************************************************************

                                 ; Miscellaneous commands

034776  777777  777655           .DBCMD: -123                           ; ignore
034777  50 45 54 60 00 00                SIXBIT  /HELP/                 ; print help message
035000  45 70 51 64 00 00                SIXBIT  /EXIT/                 ; exit DEBUG mode
035001  44 44 64 00 00 00                SIXBIT  /DDT/                  ; enter DDT

035002  63 45 64 00 00 00                SIXBIT  /SET/                  ; set/print options
035003  64 41 53 45 00 00                SIXBIT  /TAKE/                 ; take commands from a file
035004  56 64 41 53 45 00                SIXBIT  /NTAKE/                ; take commands from a file
035005  45 56 45 70 64 00                SIXBIT  /ENEXT/                ; examine next location
035006  44 56 45 70 64 00                SIXBIT  /DNEXT/                ; examine next location
035007  43 57 56 46 51 47                SIXBIT  /CONFIG/               ; determine/print configuration
035010  43 57 56 60 56 64                SIXBIT  /CONPNT/               ; print configuration
035011  63 45 54 45 43 64                SIXBIT  /SELECT/               ; select device to test

                                 ; CSR commands

035012  45 43 63 62 00 00                SIXBIT  /ECSR/                 ; examine CSR register
035013  44 43 63 62 00 00                SIXBIT  /DCSR/                 ; deposit CSR register
035014  72 43 63 62 00 00                SIXBIT  /ZCSR/                 ; zero CSR register

035015  62 45 63 45 64 00                SIXBIT  /RESET/                ; issue an EBUS Reset
035016  43 54 45 41 62 00                SIXBIT  /CLEAR/                ; issue a 'Port Clear'

                                 ; Start/stop/single step commands

035017  63 64 41 62 64 00                SIXBIT  /START/                ; start the port
035020  63 64 57 60 00 00                SIXBIT  /STOP/                 ; stop the port
035021  43 57 56 64 00 00                SIXBIT  /CONT/                 ; continue the port
035022  63 63 64 45 60 00                SIXBIT  /SSTEP/                ; single step the port
035023  63 64 62 41 43 45                SIXBIT  /STRACE/               ; single step the port while tracing
035024  63 60 62 51 56 64                SIXBIT  /SPRINT/               ; print single step history
035025  63 43 54 45 41 62                SIXBIT  /SCLEAR/               ; clear single step history data
035026  63 63 51 56 51 64                SIXBIT  /SSINIT/               ; set up ss start addr/data/history

                                 ; EBUF commands

035027  45 45 42 65 46 00                SIXBIT  /EEBUF/                ; examine EBUF register
035030  44 45 42 65 46 00                SIXBIT  /DEBUF/                ; deposit EBUF register
035031  72 45 42 65 46 00                SIXBIT  /ZEBUF/                ; zero EBUF register

                                 ; RAR/LAR commands

035032  45 54 41 62 00 00                SIXBIT  /ELAR/                 ; examine LAR register
035033  44 62 41 62 00 00                SIXBIT  /DRAR/                 ; deposit RAR register
035034  72 62 41 62 00 00                SIXBIT  /ZRAR/                 ; zero RAR register
```

```


                                        ; CRAM commands

       035035  44 43 62 41 55 00                 SIXBIT  /DCRAM/                 ; deposit into port CRAM
       035036  42 43 62 41 55 00                 SIXBIT  /BCRAM/                 ; deposit into port CRAM by bit field
       035037  41 43 62 41 55 00                 SIXBIT  /ACRAM/                 ; alter port CRAM
       035040  45 43 62 41 55 00                 SIXBIT  /ECRAM/                 ; examine port CRAM
       035041  54 43 62 41 55 00                 SIXBIT  /LCRAM/                 ; list port CRAM
       035042  72 43 62 41 55 00                 SIXBIT  /ZCRAM/                 ; zero port CRAM

                                        ; 2901 command

       035043  45 22 31 20 21 00                 SIXBIT  /E2901/                 ; examine 2901 registers
       035044  44 22 31 20 21 00                 SIXBIT  /D2901/                 ; deposit 2901 registers
       035045  72 22 31 20 21 00                 SIXBIT  /Z2901/                 ; zero 2901 registers

                                        ; Local storage commmands

       035046  45 54 57 43 63 00                 SIXBIT  /ELOCS/                 ; examine local storage
       035047  44 54 57 43 63 00                 SIXBIT  /DLOCS/                 ; deposit local storage
       035050  72 54 57 43 63 00                 SIXBIT  /ZLOCS/                 ; zero local storage

                                        ; Scope looping commands

       035051  54 43 57 56 51 00                 SIXBIT  /LCONI/                 ; loop on CONI
       035052  54 43 57 56 57 00                 SIXBIT  /LCONO/                 ; loop on CONO
       035053  54 44 41 64 41 51                 SIXBIT  /LDATAI/                ; loop on DATAI
       035054  54 44 41 64 41 57                 SIXBIT  /LDATAO/                ; loop on DATAO
       035055  54 62 57 65 64 56                 SIXBIT  /LROUTN/                ; loop on routine

                                        ; Mark bit commands

       035056  55 41 62 53 00 00                 SIXBIT  /MARK/                  ; set mark bit in locations
       035057  62 55 41 62 53 00                 SIXBIT  /RMARK/                 ; remove mark bit in locations
       035060  43 55 41 62 53 00                 SIXBIT  /CMARK/                 ; clear mark bit in locations
       035061  54 55 41 62 53 00                 SIXBIT  /LMARK/                 ; list locations with mark bit set

                                        ; Breakpoint commands

       035062  42 62 45 41 53 00                 SIXBIT  /BREAK/                 ; set breakpoint
       035063  62 42 62 45 41 53                 SIXBIT  /RBREAK/                ; remove breakpoint
       035064  43 42 62 45 41 53                 SIXBIT  /CBREAK/                ; clear all breakpoints
       035065  54 42 62 45 41 53                 SIXBIT  /LBREAK/                ; list breakpoints

                                        ; Microcode file commands

       035066  54 57 41 44 00 00                 SIXBIT  /LOAD/                  ; load microcode from test into CRAM
       035067  54 51 63 64 00 00                 SIXBIT  /LIST/                  ; list microcode from test
       035070  46 54 57 41 44 00                 SIXBIT  /FLOAD/                 ; load microcode from file into CRAM
       035071  46 66 45 62 46 71                 SIXBIT  /FVERFY/                ; verify microcode loaded
       035072  46 54 51 63 64 00                 SIXBIT  /FLIST/                 ; list microcode in file
       035073  46 45 70 41 55 00                 SIXBIT  /FEXAM/                 ; examine microcode in file
```

```
                                        ;#**************************************************************
                                        ;* DSETLN - Set length of data transfer
                                        ;#**************************************************************

035542  210 01 0 00 057177              DSETLN: MOVN    1,[FMSGCD (SET LEN length (1-102400.) or SET LEN <CR>)]
035543  260 17 0 00 033057*                     GO      .DARG                   ; get argument
035544  263 17 0 00 000000                        RTN                           ; error/altmode/question - exit
035545  254 00 0 00 035554                        JRST  DSETL1                  ; no argument given
035546  200 00 0 00 035471*                     MOVE    ARGUM                   ; get argument
035547  303 00 0 00 000000                      CAILE   0                       ; argument in range 1-102400?
035550  303 00 0 00 310000                      CAILE   ^D102400
                                                JRST    [FMSGCD <? Out of range 1-102400. words>
                                                         GO      FIOFF
035551  254 00 0 00 057207                               RTN]
035552  202 00 0 00 035766                      MOVEM   DSWORD                  ; save it
035553  263 17 0 00 000000                      RTN                             ; exit

035554  037 00 0 00 030242              DSETL1: PCRL
035555  037 00 0 00 057212              DSETL2: TMSG    <Buffer length (words) ..... >
035556  200 00 0 00 035766                      MOVE    DSWORD
035557  037 15 0 00 000000                      PNTDEC
035560  037 00 0 00 055243                      TMSGD   <.>
035561  263 17 0 00 000000                      RTN                             ; exit


                                        ;#**************************************************************
                                        ;* DSETDA - Set data pattern to insert in buffer
                                        ;#**************************************************************

035562  200 01 0 00 057235              DSETDA: MOVE    1,[FMSGCD (SET PAT number (1-126 octal)(126=opr sel) or SET PAT <CR>)]
035563  260 17 0 00 035466*                     GO      .OARG                   ; get argument
035564  263 17 0 00 000000                        RTN                           ; error/altmode/question - exit
035565  254 00 0 00 035607                        JRST  DSETD1                  ; no argument given
035566  200 00 0 00 035546*                     MOVE    ARGUM                   ; get argument
035567  301 00 0 00 000001                      CAIL    1                       ; in range?
035570  303 00 0 00 000126                      CAILE   126
                                                JRST    [FMSGCD <? Out of range 1-126 octal>
                                                         GO      FIOFF
035571  254 00 0 00 057245                               RTN]
035572  202 00 0 00 035767                      MOVEM   DSPAT                   ; save it
035573  302 00 0 00 000126                      CAIE    126                     ; operator selected pattern?
035574  254 00 0 00 035603                      JRST    DSETDX                  ; no - exit
035575  200 01 0 00 057256                      MOVE    1,[FMSGCD <SET PAT 126,data (CR)>]
035576  260 17 0 00 035563*                     GO      .OARG                   ; get argument
035577  263 17 0 00 000000                        RTN                           ; error/altmode/question - exit
                                                  JRST  [FMSGCD <? Missing argument>
                                                         GO      FIOFF
035600  254 00 0 00 055164                               RTN]
```

```
035601  200 00 0 00 035566*            MOVE    ARGUM                          ; get argument
035602  202 00 0 00 035770             MOVEM   DSTPAT                         ; save it
035603  260 17 0 00 000000*    DSETDX: GO      LASARG                         ; check for last argument
035604  263 17 0 00 000000             RTN                                    ; exit
035605  037 01 0 00 057250             FMSGCD  <SET PAT 126,data (CR)>
035606  263 17 0 00 000000             RTN                                    ; exit

035607  037 00 0 00 030242     DSETD1: PCRL
035610  037 00 0 00 057257     DSETD2: TMSG    <Data pattern (#) .......... >
035611  200 01 0 00 035767             MOVE    1,DSPAT                        ; get pattern
035612  256 00 0 01 000000*            XCT     PATPNT(1)                      ;   ...
035613  302 01 0 00 000126             CAIE    1,126                          ; operator selected pattern?
035614  254 00 0 00 035620             JRST    DSETD3                         ; no - continue
035615  037 00 0 00 054243             TMSG    < - >                          ; yes - print it out
035616  200 00 0 00 035770             MOVE    DSTPAT
035617  037 13 0 00 000000             PNTHW
035620  037 00 0 00 030242     DSETD3: PCRL
035621  263 17 0 00 000000             RTN                                    ; exit



                               ;#*********************************************************************
                               ;* DSETFN - Set default microcode file name
                               ;#*********************************************************************

035622  200 01 0 00 057277     DSETFN: MOVE    1,[FMSGCD (SET FNAME file ext <CR> or SET FNAME <CR>)]
035623  260 17 0 00 035523*            GO      .SARG                          ; get argument (file name)
035624  263 17 0 00 000000              RTN                                   ; error/altmode/question - exit
035625  254 00 0 00 035641              JRST   DSETF1                         ; no argument - print it out
035626  200 00 0 00 035601*            MOVE    ARGUM                          ; get argument
035627  202 00 0 00 064645'            MOVEM   TMP1                           ; save it
035630  210 01 0 00 057277             MOVN    1,[FMSGCD (SET FNAME file ext <CR> or SET FNAME <CR>)]
035631  260 17 0 00 035623*            GO      .SARG                          ; get argument (extension)
035632  263 17 0 00 000000              RTN                                   ; error/altmode/question - exit
035633  254 00 0 00 035636              JRST   DSETF0                         ; no extension given - exit
035634  200 00 0 00 035626*            MOVE    ARGUM                          ; get 2nd argument
035635  202 00 0 00 000000#            MOVEM   UNAME+1                        ; save it
035636  200 00 0 00 064645'    DSETF0: MOVE    TMP1                           ; get first argument
035637  202 00 0 00 000000*            MOVEM   UNAME                          ; save it
035640  263 17 0 00 000000             RTN                                    ; return

035641  037 00 0 00 030242     DSETF1: PCRL
035642  037 00 0 00 057300     DSETF2: TMSG    <Microcode file name ....... >
035643  200 00 0 00 035637*            MOVE    UNAME                          ; get name
035644  037 00 0 00 000002             PNTSIX                                 ; print it
035645  037 00 0 00 057306             TMSG    <.>
035646  200 00 0 00 000000#            MOVE    UNAME+1                        ; get extension
035647  037 00 0 00 000002             PNTSIX                                 ; print it
035650  037 00 0 00 030242             PCRL
035651  263 17 0 00 000000             RTN                                    ; return
```

```
036223  260 17 0 00 000000*             GO      IPASST                  ; single step once
                                         JRST   [FMSGCD <? Error accessing CSR register>
                                                 GO     FIOFF
036224  254 00 0 00 057707                       RTN]
                                         JRST   [FMSGCD <? Port not stopped>
                                                 GO     FIOFF
036225  254 00 0 00 057717                       RTN]
                                         JRST   [GO     SAVHST
                                                 TMSGC  <[Error bits set in CSR]>
                                                 SKIPE  DTRFLG
                                                 JRST   DSSTX
                                                 TMSGC  <[>
                                                 MOVE   4
                                                 PNTOCS
                                                 TMSGD  < ssteps done]>
036226  254 00 0 00 060034                       JRST   DSSTX]
036227  260 17 0 00 037523              GO      SAVHST                  ; save data for sstep history

                                ; If tracing is being done

036230  200 01 0 00 036303              MOVE    1,SSLOC                 ; get history location
036231  332 00 0 00 064577'             SKIPE   DTRFLG                  ; tracing being done?
036232  260 17 0 00 037504              GO      PNTHST                  ; yes - print it

                                ; Altmode and exit checking

036233  037 07 0 00 000003      DSST1:  TTALTM                          ; altmode key struck?
036234  334 00 0 00 000000              SKIPA                           ; no - proceed
                                        JRST    [PCRL                   ; yes - print some final
                                                 SKIPE  DTRFLG          ;     data and exit
                                                 RTN
                                                 TMSG   <[>
                                                 MOVE   4
                                                 PNTOCS
                                                 TMSGD  < ssteps done]>
036235  254 00 0 00 060046                       JRST   DSSTX]
036236  367 02 0 00 036217              SOJG    2,DSST0                 ; loop till done

                                ; Exit

036237  037 00 0 00 060056      DSSTX:  TMSGC   <LAR Last/>             ; print LAR data
036240  200 00 0 00 036142*             MOVE    SLAST                   ; get last address
036241  037 04 0 00 000000              PNT4                            ; print it
036242  037 00 0 00 060061              TMSG    <  Next/>
036243  200 00 0 00 036170*             MOVE    SNEXT                   ; get next address
036244  037 04 0 00 000000              PNT4                            ; print it
036245  037 00 0 00 060063              TMSG    <  CSR/>                ; print CSR data
036246  200 00 0 00 036221*             MOVE    SDATA                   ; get it
036247  037 13 0 00 000000              PNTHW                           ; print it
036250  037 00 0 00 030242              PCRL                            ; end of line
036251  263 17 0 00 000000              RTN
```

```

                                ;#*********************************************************************
                                ;* DSSPNT - Print single step history
                                ;#*********************************************************************

036252  201 00 0 00 000024      DSSPNT: MOVEI   ^D20                    ; set default number of ssteps
036253  202 00 0 00 036213*             MOVEM   ARGUM                   ;   to print to 20
036254  210 01 0 00 060075              MOVN    1,[FMSGCD (SPRINT EntryCount or SPRINT <CR>)]
036255  260 17 0 00 036207*             GO      .DARG                   ; get argument
036256  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
036257  255 00 0 00 000000               JFCL                           ; no argument given
036260  200 03 0 00 036253*             MOVE    3,ARGUM                 ; get argument
036261  200 01 0 00 036303              MOVE    1,SSLOC                 ; get current location
036262  037 00 0 00 060012              TMSGCD  < RAR    LAR    CSR (final)  SS#>
036263  260 17 0 00 037504      DSSPNO: GO      PNTHST                  ; print an item
036264  037 07 0 00 000003              TTALTM                          ; altmode typed?
036265  334 00 0 00 000000              SKIPA                           ; no - continue
036266  263 17 0 00 000000              RTN                             ; yes - exit
036267  375 00 0 00 000001              SOSGE   1                       ; decrement pointer - at start of list?
036270  201 01 0 00 000177              MOVEI   1,177                   ; yes - point to end now
036271  333 00 0 01 036304              SKIPLE  SSNUM(1)                ; valid entry?
036272  367 03 0 00 036263              SOJG    3,DSSPNO                ; yes - loop till done
036273  263 17 0 00 000000              RTN                             ; no - return


                                ;#*********************************************************************
                                ;* DSSCLR - Clear single step history data
                                ;#*********************************************************************

036274  200 01 0 00 060102      DSSCLR: MOVE    1,[FMSGCD (SCLEAR <CR>)]
036275  260 17 0 00 036161*             GO      CHKARG                  ; check for argument
036276  263 17 0 00 000000              RTN                             ; error - exit
036277  402 00 0 00 036303              SETZM   SSLOC                   ; clear first word
036300  200 00 0 00 060103              MOVE    [SSLOC,,SSLOC+1]        ; build BLT pointer
036301  251 00 0 00 037503              BLT     SSCSRN+177              ; clear it
036302  263 17 0 00 000000              RTN                             ; return

                                ; Ring buffer and miscellaneous data

036303  000000  000000          SSLOC:  0                               ; pointer to single step entry
036304                          SSNUM:  BLOCK   200                     ; latest single step number
036504                          SSADRL: BLOCK   200                     ; initial address
036704                          SSADRN: BLOCK   200                     ; final address
037104                          SSCSRL: BLOCK   200                     ; initial CSR
037304                          SSCSRN: BLOCK   200                     ; final CSR
```

```

                                         ; Unrecognizable - check if a field was specified

040130  200 01 0 00 064636'                      MOVE    1,SAVFLD                ; get command into AC1
040131  201 02 0 00 040243                       MOVEI   2,FLDLIS                ; get address of field list
040132  260 17 0 00 035411*                      GO      .COMM                   ; handle command decoding
                                                  JRST   [FMSGCD <? Argument error>
                                                          GO      CLRBUF
                                                          PCRLF
                                                          GO      FIOFF
040133  254 00 0 00 060314                                JRST    DCOMA2]        ; no - error - reask question
040134  370 00 0 00 000001                       SOS     1                       ; normalize to -1..12
040135  200 06 0 00 000001                       MOVE    6,1                     ; put into AC6
040136  254 00 0 00 040074                       JRST    DCOMA1                  ; go handle the field

                                         ; Insert field entry

040137  200 00 0 00 000001               DCOMA4: MOVE    1                       ; get number
040140  260 17 1 06 040221                       GO      @DCODPB(6)              ; yes - insert it
040141  200 00 0 00 030231               DCOMA5: MOVE    $TTCHR                  ; get last character typed
040142  312 00 0 00 000015                       CAME    15                      ; CRLF?
040143  254 00 0 00 040074                       JRST    DCOMA1                  ; no - get next entry

                                         ; Finally, write the location

040144  260 17 0 00 040376               DCOMAX: GO      DWCRAM                  ; go write CRAM location
040145  350 01 0 00 037746                       AOS     1,CRAMFR                ; update the specified address
040146  202 01 0 00 037755                       MOVEM   1,ANEXT                 ; save next address
040147  313 01 0 00 037747                       CAMLE   1,CRAMTO                ; reach end yet?
040150  263 17 0 00 000000                       RTN                             ; yes - exit
040151  200 00 0 00 064576'                      MOVE    DFLAG                   ; get dispatch flag
040152  302 00 0 00 000003                       CAIE    3                       ; alter?
040153  254 00 0 00 040045                       JRST    DCOMDB                  ; no - deposit by bit field
040154  254 00 0 00 040052                       JRST    DCOMAL                  ; yes - alter by bit field
```

```
 4701
 4702                                    ; Input message table
 4703
 4704  040155  037 01 0 00 060325        DCOMSG: FMSG    <J - >
 4705  040156  037 01 0 00 060326                FMSG    <PAR - >
 4706  040157  037 01 0 00 060330                FMSG    <DENA - >
 4707  040160  037 01 0 00 060332                FMSG    <MGC - >
 4708  040161  037 01 0 00 060334                FMSG    <SORC - >
 4709  040162  037 01 0 00 060336                FMSG    <FUNC - >
 4710  040163  037 01 0 00 060340                FMSG    <DEST - >
 4711  040164  037 01 0 00 060342                FMSG    <CENA - >
 4712  040165  037 01 0 00 060344                FMSG    <RAM - >
 4713  040166  037 01 0 00 060346                FMSG    <A - >
 4714  040167  037 01 0 00 060347                FMSG    <B - >
 4715  040170  037 01 0 00 060350                FMSG    <SK - >
 4716  040171  037 01 0 00 060352                FMSG    <BUS - >
 4717  040172  037 01 0 00 060354                FMSG    <CRY - >
 4718  040173  037 01 0 00 060356                FMSG    <CTL - >
 4719  040174  037 01 0 00 060360                FMSG    <TIME - >
 4720  040175  037 01 0 00 060362                FMSG    <SPARE - >
 4721  040176  037 01 0 00 060364                FMSG    <MARK - >
 4722
 4723                                    ; Microword load byte pointer table
 4724
 4725                                    DCOLDB: [LDB    [POINT 12,CWORDL,17]    ; J - Bits 6-17
 4726  040177  000000  060367                     RTN]
 4727                                            [LDB    [POINT 1,CWORDL,18]     ; PAR - Bit 18
 4728  040200  000000  060372                     RTN]
 4729                                            [LDB    [POINT 1,CWORDL,19]     ; OENA - Bit 19
 4730  040201  000000  060375                     RTN]
 4731                                            [LDB    [POINT 10,CWORDL,29]    ; MGC - Bits 20-29
 4732  040202  000000  060400                     RTN]
 4733                                            [LDB    [POINT 3,CWORDL,32]     ; SORC - Bits 30-32
 4734  040203  000000  060403                     RTN]
 4735                                            [LDB    [POINT 3,CWORDL,35]     ; FUNC - Bits 33-35
 4736  040204  000000  060406                     RTN]
 4737                                            [LDB    [POINT 3,CWORDR,8]      ; DEST - Bits 6-8
 4738  040205  000000  060411                     RTN]
 4739                                            [LDB    [POINT 1,CWORDR,9]      ; CENA - Bit 9
 4740  040206  000000  060414                     RTN]
 4741                                            [LDB    [POINT 1,CWORDR,10]     ; RAM - Bit 10
 4742  040207  000000  060417                     RTN]
 4743                                            [LDB    [POINT 4,CWORDR,14]     ; A - Bits 11-14
 4744  040210  000000  060422                     RTN]
 4745                                            [LDB    [POINT 4,CWORDR,18]     ; B - Bits 15-18
 4746  040211  000000  060425                     RTN]
 4747                                            [LDB    [POINT 5,CWORDR,23]     ; SK - Bits 19-23
 4748  040212  000000  060430                     RTN]
 4749                                            [LDB    [POINT 3,CWORDR,26]     ; BUS - Bits 24-26
 4750  040213  000000  060433                     RTN]
```

```
                                ; Save CRAM location

040677  200 01 0 00 061154              MOVE    1,[-1,,7750]            ; set up AOBJN word
040700  260 17 0 00 044040              GO      SAVCRM                  ; save the CRAM location

                                ; Now do each location

040701  200 02 0 00 064626'     DZALU1: MOVE    2,RALUB                 ; get location to write
040702  137 02 0 00 061203              DPB     2,[POINT 4,DZALUM+1,18] ; set up register number
040703  201 00 0 00 000002              MOVEI   2                       ; get destination field of 2 (BA)
040704  306 02 0 00 000020              CAIN    2,20                    ; Q-Register?
040705  201 00 0 00 000000              MOVEI   0                       ; yes - use destination of 0 (QF)
040706  137 00 0 00 061204              DPB     [POINT 3,DZALUM+1,8]    ; set up destination field
040707  201 01 0 00 040750              MOVEI   1,DZALUM                ; set up microcode address
040710  260 17 0 00 043207              GO      MLOADN                  ; load/verify it
040711  255 00 0 00 000000              JFCL                            ; ignore error
040712  350 00 0 00 000002              AOS     2                       ; point to next register
040713  202 02 0 00 037755              MOVEM   2,ANEXT                 ; save next register
040714  201 01 0 00 007750              MOVEI   1,7750                  ; get start address
040715  242 01 0 00 000001              LSH     1,1                     ; position correctly
040716  260 17 0 00 040621*             GO      LDRAR                   ; load the RAR
040717  201 01 0 00 020010              MOVEI   1,MPRUN!SINCYC          ; start it up
040720  260 17 0 00 040627*             GO      LDCSR
040721  037 07 0 00 000003              TTALTM                          ; altmode key struck?
040722  334 00 0 00 000000              SKIPA                           ; no - proceed
040723  254 00 0 00 040727              JRST    DZALUX                  ; yes - exit
040724  350 01 0 00 064626'             AOS     1,RALUB                 ; point to next location
040725  317 01 0 00 064627'             CAMG    1,RALUE                 ; done yet?
040726  254 00 0 00 040701              JRST    DZALU1                  ; no - loop till done

                                ; Restore CRAM location and exit

040727  200 01 0 00 061154      DZALUX: MOVE    1,[-1,,7750]            ; set up AOBJN word
040730  260 17 0 00 044062              GO      RESCRM                  ; restore CRAM data
040731  263 17 0 00 000000              RTN                             ; return
```

```


                                ; Microcode to examine 2901's

040732  775077  502004          DEALUM: MWORD   <ADDR=7750,JMAP,J=7750,SOA,OR,D=1,OENA,SELE,MGC=4>
040734  777777  777777                   -1

                                ; Microcode to deposit 2901's

040735  775077  510000          DDALUM: MWORD   <ADDR=7750,JMAP,J=7751,D=1>             ; 7750
040737  775177  530000                  MWORD   <CJP,J=7753,D=1,CENA,CCER>              ; 7751
040741  775277  510000                  MWORD   <JMAP,J=7751,D=1>                       ; 7752
040743  775300  000010                  MWORD   <CONT,SDO,OR,D=2,SELE,MGC=10>           ; 7753
040745  775477  540000                  MWORD   <JMAP,J=7754,D=1>                       ; 7754
040747  777777  777777                  -1

                                ; Microcode to zero 2901's

040750  775077  500000          DZALUM: MWORD   <ADDR=7750,JMAP,J=7750,SOA,AND,D=2>
040752  777777  777777                  -1

                                ; Arguments

040753  000000  000000          ALUFR:  0                               ; default range of
040754  000000  000020          ALUTO:  20                              ;   registers
```

```
                                        ;#*****************************************************************
                                        ;* DLMARK - List locations with mark bit set
                                        ;
                                        ; If arguments are given, they are assumed to the the locations between
                                        ; which all mark bits are to be listed.
                                        ;#*****************************************************************

                                        ; Start to process the arguments typed - both should be CRAM
                                        ; addresses in octal.

041377  400 05 0 00 000000              DLMARK: SETZ    5,                      ; init range of addresses
041400  201 06 0 00 007777                      MOVEI   6,7777                  ;   to 0-7777

                                        ; Get first address

041401  200 01 0 00 061337                      MOVE    1,[FMSGCD <LMARK addr1,addr2 (CR)>]
041402  260 17 0 00 041340*                     GO      .OARG                   ; get first address
041403  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
041404  254 00 0 00 041416                       JRST   DLMARO                  ; no arguments given - continue
041405  200 00 0 00 041343*                     MOVE    ARGUM                   ; get argument
041406  202 00 0 00 000005                      MOVEM   5                       ; save in 'from' location
041407  202 00 0 00 000006                      MOVEM   6                       ; save in 'to' location

                                        ; Get 2nd address

041410  210 01 0 00 061337                      MOVN    1,[FMSGCD <LMARK addr1,addr2 (CR)>]
041411  260 17 0 00 041402*                     GO      .OARG                   ; get first address
041412  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
041413  254 00 0 00 041416                       JRST   DLMARO                  ; no arguments given - continue
041414  200 00 0 00 041405*                     MOVE    ARGUM                   ; get argument
041415  202 00 0 00 000006                      MOVEM   6                       ; save in 'to' location

                                        ; Now range check the arguments

041416  331 00 0 00 000005              DLMARO: SKIPL   5                       ; 1st argument in range?
041417  303 05 0 00 007777                      CAILE   5,7777
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO      FIOFF
041420  254 00 0 00 057070                               RTN]
041421  331 01 0 00 000006                      SKIPL   1,6                     ; 2nd argument in range?
041422  303 06 0 00 007777                      CAILE   6,7777
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO      FIOFF
041423  254 00 0 00 057070                               RTN]
041424  274 01 0 00 000005                      SUB     1,5                     ; see if range ok
                                                JUMPL   1,[FMSGCD <? Range nonsensical>
                                                           GO    FIOFF
041425  321 01 0 00 057436                                 RTN]
```

```

                                     ; Then ensure the port is stopped

041426  260 17 0 00 041355*                 GO      ISTOP                  ; stop the port
041427  200 07 0 00 061340                  MOVE    7,[-2]                 ; initialize mark bit count
041430  037 00 0 00 061341                  TMSGC   <Mark bit set at:  >

                                     ; Now obtain CRAM data and check mark bit

041431  202 05 0 00 037750           DLMAR1: MOVEM  5,CADDR                ; set up address to load
041432  476 00 0 00 037751                  SETOM   CWORDL                 ; initialize left half
041433  476 00 0 00 037752                  SETOM   CWORDR                 ; initialize right half
041434  260 17 0 00 040430                  GO      DRCRAM                 ; read the data
041435  200 03 0 00 037752                  MOVE    3,CWORDR               ; get right word
041436  606 03 0 00 000001                  TRNN    3,1                    ; mark bit set?
041437  254 00 0 00 041445                  JRST    DLMAR2                 ; no - continue

                                     ; Print location

041440  350 00 0 00 000007                  AOS     7                      ; next 5 characters
041441  303 07 0 00 000007                  CAILE   7,7                    ; done with line?
                                            GO      [SETO   7,             ; yes - init counter
                                                     TMSGC  <
                                                    >
041442  260 17 0 00 061353                           RTN]
041443  001 04 0 00 000005                  PNTOCC  4,5                    ; print address
041444  037 00 0 00 000040                  PSP                            ; print a space

                                     ; Increment to next location

041445  037 07 0 00 000003           DLMAR2: TTALTM                        ; altmode key struck?
041446  334 00 0 00 000000                  SKIPA                          ; no - proceed
041447  254 00 0 00 041453                  JRST    DLMARX                 ; yes - exit
041450  350 00 0 00 000005                  AOS     5                      ; increment 'from' address
041451  317 05 0 00 000006                  CAMG    5,6                    ; exceed 'to' address?
041452  254 00 0 00 041431                  JRST    DLMAR1                 ; no - get next address
041453  037 00 0 00 030242           DLMARX: PCRL                          ; yes - print blank line
041454  263 17 0 00 000000                  RTN                            ;   and exit
```

```

                                                  ;#*****************************************************************
                                                  ;* DLIST - List microcode from test
                                                  ;#*****************************************************************

042062  210 01 0 00 061563          DLIST:  MOVN    1,[FMSGCD (LIST XXnnn <CR>)]
042063  260 17 0 00 042040*                 GO      .SARG                   ; get argument
042064  263 17 0 00 000000                   RTN                            ; error/altmode/question - exit
                                             JRST   [FMSGCD <? Missing argument>
                                                     GO      FIOFF
042065  254 00 0 00 055164                           RTN]
042066  200 01 0 00 042043*                 MOVE    1,ARGUM                 ; get argument
042067  260 17 0 00 034315                  GO      TSTDEC                  ; check for alternate format
                                             JRST   [FMSGCD <? Argument error>
                                                     GO      FIOFF
042070  254 00 0 00 061517                           RTN]
                                             JRST   [FMSGC  <? Test number out of range 1 - >
                                                     MOVE    1
                                                     PNTOCF
                                                     PCRLF
                                                     GO      FIOFF
042071  254 00 0 00 061522                           RTN]
042072  260 17 0 00 042775                  GO      TSTGET                  ; get test data
042073  200 01 0 00 047016                  MOVE    1,TSTMIC                ; get test microcode address
                                            JUMPLE  1,[FMSGCD <? Test specified does not have microcode>
                                                       GO      FIOFF
042074  323 01 0 00 061541                             RTN]
042075  260 17 0 00 043446                  GO      MLIST                   ; list it
042076  263 17 0 00 000000                  RTN                             ; continue
```

```
                                        ;#*********************************************************************
                                        ;* DFLOAD - Load microcode from file into CRAM
                                        ;#*********************************************************************

042077  200 01 0 00 061567      DFLOAD: MOVE    1,[FMSGCD (FLOAD <CR>)]
042100  260 17 0 00 041575*             GO      CHKARG                  ; check for argument
042101  263 17 0 00 000000              RTN                             ; error - exit
042102  400 02 0 00 000000              SETZ    2,                      ; init count of words loaded
042103  260 17 0 00 000000*             GO      FINIT                   ; init file for reading
042104  263 17 0 00 000000               RTN                            ; failed - return
042105  260 17 0 00 042052*             GO      ISTOP                   ; stop the port
042106  260 17 0 00 000000*     DFLOA0: GO      FGETW                   ; get microcode entry
042107  254 00 0 00 042121               JRST   DFLOAX                  ; error/EOF occurred
042110  261 17 0 00 035764              PUT     PARFLG                  ; save state of parity flag
042111  402 00 0 00 035764              SETZM   PARFLG                  ; ensure no parity is generated
042112  260 17 0 00 040376              GO      DWCRAM                  ; load it
042113  262 17 0 00 035764              GET     PARFLG                  ; restore state of parity flag
042114  350 00 0 00 000002              AOS     2                       ; increment count
042115  332 00 0 00 064570'             SKIPE   ALTF                    ; altmode typed?
042116  254 00 0 00 042121              JRST    DFLOAX                  ; yes - exit
042117  037 07 0 00 000003              TTALTM                          ; altmode key struck?
042120  254 00 0 00 042106              JRST    DFLOA0                  ; no - loop til done

042121  037 00 0 00 061570      DFLOAX: TMSGC   <[Number of CRAM locations loaded = >
042122  200 00 0 00 000002              MOVE    2
042123  037 15 0 00 000000              PNTDEC
042124  037 00 0 00 061556              TMSGD   <.]>
042125  037 00 0 00 061600              TMSGC   <[Ucode version >
042126  201 00 0 00 000137              MOVEI   137                     ; read location 137 which contains
042127  202 00 0 00 037750              MOVEM   CADDR                   ;   microcode version in MGC field
042130  260 17 0 00 040430              GO      DRCRAM                  ;   bits 14-23
042131  135 00 0 00 060377              LDB     [POINT 10,CWORDL,29]
042132  037 16 0 00 000003              PNTOCS                          ; print it
042133  037 00 0 00 054523              TMSGD   <]>
042134  263 17 0 00 000000              RTN                             ; return
```

```
                                                SUBTTL  Program Utility Routines

                                        ;#*********************************************************************
                                        ;* TRACE - Program Trace Routine
                                        ;
                                        ; This routine traces test execution if the TRACE switch is set.  This
                                        ; routine sets up TSTSUB, ERFLG, ERRPC for error reporting during  the
                                        ; test, and ERRNUM for interrupt handling.
                                        ;#*********************************************************************

        042717  261 17 0 00 000000      TRACE:  RPUT    (0,1)                   ; save AC's
        042721  202 00 0 00 047026              MOVEM   TSTOFF                  ; save relocatable offset
        042722  260 17 0 00 034410*             GO      SWITT                   ; get switches
        042723  606 00 0 00 400000              TRNN    TRACET                  ; want to trace?
        042724  254 00 0 00 042743              JRST    TRACE0                  ; no

                                        ; Print test number if tracing

        042725  037 00 0 00 062162              TMSG    <* >
        042726  260 17 0 00 033310*             GO      PNTDEV                  ; print device name
        042727  037 00 0 00 062163              TMSG    < Test >
        042730  200 01 0 00 047010              MOVE    1,TSTCLS                ; get test class
        042731  200 00 0 01 031275              MOVE    INDLIS(1)               ; get class description
        042732  037 00 0 00 000002              PNTSIX                          ; print it
        042733  200 00 0 00 047011              MOVE    TSTNUM                  ; get test number
        042734  037 16 0 00 000003              PNTOCS                          ; print it
        042735  200 00 0 00 047014              MOVE    TSTFLG                  ; get test flags
        042736  603 00 0 00 000100              TLNE    (TDENA)                 ; disabled?
        042737  037 00 0 00 062165              TMSG    < (Disabled)>           ; yes - print such
        042740  603 00 0 00 001000              TLNE    (TUSER)                 ; run in user mode?
        042741  037 00 0 00 062170              TMSG    < (Exec only test)>     ; no - print such
        042742  037 00 0 00 030242              PCRL

                                        ; Initialize flags

        042743  402 00 0 00 000015      TRACE0: SETZM   ERFLG                   ; clear the error detected flag
        042744  402 00 0 00 030052              SETZM   ERRPC                   ; clear the error PC
        042745  402 00 0 00 047012              SETZM   TSTSUB                  ; clear subtest number
        042746  402 00 0 00 000000*             SETZM   SCOOFF                  ; clear error offset
        042747  476 00 0 00 042657*             SETOM   INTNUM                  ; initialize interrupt number
        042750  402 00 0 00 041623*             SETZM   SNEXT                   ; initialize next address
        042751  402 00 0 00 041613*             SETZM   SDATA                   ; initialize CSR data
```

```
                                        ; Determine whether or not test should be run

042752  200 00 0 00 047014                      MOVE    TSTFLG                  ; get test flags
042753  603 00 0 00 001100                      TLNE    (TDENA!TUSER)           ; test disabled or exec mode only?
042754  254 00 0 00 042771                      JRST    TRACEX                  ; yes - exit test

                                        ; Handle microcode loaded flags

042755  316 16 0 00 047001                      CAMN    MBCN,PORTNI             ; NI port selected?
042756  332 00 0 00 047016                      SKIPE   TSTMIC                  ; yes - any test microcode?
042757  334 00 0 00 000000                      SKIPA                           ; no - continue
042760  476 00 0 00 064646'                     SETOM   TSLOD1                  ; yes - init ucode loaded flag (NI)
042761  316 16 0 00 047002                      CAMN    MBCN,PORTCI             ; CI port selected?
042762  332 00 0 00 047016                      SKIPE   TSTMIC                  ; yes - any test microcode?
042763  334 00 0 00 000000                      SKIPA                           ; no - continue
042764  476 00 0 00 064647'                     SETOM   TSLOD2                  ; yes - init ucode loaded flag (CI)

                                        ; Exit and run test

042765  262 17 0 00 000001                      RGET    (1,0)                   ; restore AC's
042767  350 00 0 17 777777                      AOS     -1(P)                   ; increment test return address
042770  263 17 0 00 000000                      RTN                             ; exit normally

                                        ; Exit and skip test

042771  262 17 0 00 000001              TRACEX: RGET    (1,0,XXX)               ; restore AC's
042774  263 17 0 00 000000                      RTN                             ; exit test
```

```
                                        ;#*****************************************************************
                                        ;* PNTCRM - Print CRAM word
                                        ;
                                        ; Arguments:     AC2 - Left 30 bits
                                        ;                AC3 - Right 30 bits
                                        ;
                                        ;#*****************************************************************

043507  261 17 0 00 000000              PNTCRM: RPUT    (0,1)                   ; save AC's
043511  135 00 0 00 062233                      LDB     [POINT 12,2,17]         ; get MWJMPFLD
043512  001 04 0 00 000000                      PNTOCC  4,0                     ; print it
043513  037 00 0 00 055371                      TMSG    < >
043514  135 00 0 00 062234                      LDB     [POINT 1,2,18]          ; get MWPAR
043515  037 01 0 00 000000                      PNT1                            ; print it
043516  037 00 0 00 055371                      TMSG    < >
043517  135 00 0 00 062235                      LDB     [POINT 1,2,19]          ; get MWOUTPUTENA
043520  037 01 0 00 000000                      PNT1                            ; print it
043521  135 00 0 00 062236                      LDB     [POINT 10,2,29]         ; get MWMGCFLD
043522  001 04 0 00 000000                      PNTOCC  4,0                     ; print it
043523  037 00 0 00 055371                      TMSG    < >
043524  135 00 0 00 062237                      LDB     [POINT 3,2,32]          ; get MWSORCEFLD
043525  037 01 0 00 000000                      PNT1                            ; print it
043526  037 00 0 00 055371                      TMSG    < >
043527  135 00 0 00 062240                      LDB     [POINT 3,2,35]          ; get MWFUNCTFLD
043530  037 01 0 00 000000                      PNT1                            ; print it
043531  037 00 0 00 055371                      TMSG    < >
043532  135 00 0 00 062241                      LDB     [POINT 3,3,8]           ; get MWDESTFLD
043533  037 01 0 00 000000                      PNT1                            ; print it
043534  037 00 0 00 055371                      TMSG    < >
043535  135 00 0 00 062242                      LDB     [POINT 1,3,9]           ; get MWCCENA
043536  037 01 0 00 000000                      PNT1                            ; print it
043537  037 00 0 00 000040                      PSP
043540  135 00 0 00 062243                      LDB     [POINT 1,3,10]          ; get MWRAMODE
043541  037 01 0 00 000000                      PNT1                            ; print it
043542  037 00 0 00 000040                      PSP
043543  135 00 0 00 062244                      LDB     [POINT 4,3,14]          ; get MWPORTAFLD
043544  001 02 0 00 000000                      PNTOCC  2,0                     ; print it
043545  037 00 0 00 000040                      PSP
043546  135 00 0 00 062245                      LDB     [POINT 4,3,18]          ; get MWPORTBFLD
043547  001 02 0 00 000000                      PNTOCC  2,0                     ; print it
043550  037 00 0 00 000040                      PSP
043551  135 00 0 00 062246                      LDB     [POINT 5,3,23]          ; get MWSKIPFLD
043552  001 02 0 00 000000                      PNTOCC  2,0                     ; print it
043553  037 00 0 00 055371                      TMSG    < >
043554  135 00 0 00 062247                      LDB     [POINT 3,3,26]          ; get MWBUSCTLFLD
043555  037 01 0 00 000000                      PNT1                            ; print it
```

```
043556  037 00 0 00 055371              TMSG    < >
043557  135 00 0 00 062250              LDB     [POINT 1,3,27]          ; get MWCARRY
043560  037 01 0 00 000000              PNT1                            ; print it
043561  037 00 0 00 000040              PSP
043562  135 01 0 00 062251              LDB     1,[POINT 4,3,31]        ; get MWCTRLFLD
                                        XCT     [TMSG   <JZ  >          ; print in English
                                                 TMSG   <CJS >
                                                 TMSG   <JMAP>
                                                 TMSG   <CJP >
                                                 TMSG   <PUSH>
                                                 TMSG   <JSRP>
                                                 TMSG   <CJV >
                                                 TMSG   <JRP >
                                                 TMSG   <RFCT>
                                                 TMSG   <RPCT>
                                                 TMSG   <CRTN>
                                                 TMSG   <CJPP>
                                                 TMSG   <LDCT>
                                                 TMSG   <LOOP>
                                                 TMSG   <CONT>
043563  256 00 0 01 062272                       TMSG   <TWB >](1)
043564  037 00 0 00 055371              TMSG    < >
043565  135 00 0 00 062312              LDB     [POINT 1,3,32]          ; get MWTIMEFLD
043566  037 01 0 00 000000              PNT1                            ; print it
043567  037 00 0 00 000040              PSP
043570  135 00 0 00 062313              LDB     [POINT 2,3,34]          ; get MWSPARE00-01
043571  037 01 0 00 000000              PNT1                            ; print it
043572  037 00 0 00 000040              PSP
043573  135 00 0 00 062314              LDB     [POINT 1,3,35]          ; get MWMARKBIT
043574  037 01 0 00 000000              PNT1                            ; print it
043575  037 07 0 00 000003              TTALTM                          ; altmode typed?
043576  334 00 0 00 000000              SKIPA                           ; no - continue
                                        GO      [SETOM  ALTF            ; yes - set 'altmode typed' flag
                                                 SETZM  MULFLG          ;   and clear 'examine next' flag
043577  260 17 0 00 062315                       RTN]
043600  262 17 0 00 000001              RGET    (1,0)                   ; restore AC's
043602  263 17 0 00 000000              RTN                             ; return
```

```

                                ; Bits 3-35 - Defined per entry type.
                                ;
                                ;    Entry type 0 (SSLAST) - All zero - indicates end of table
                                ;
                                ;    Entry type 1 (SSSTRT) - Bits 3-11 - Single step address to start at
                                ;                                12-23 - Number of single steps to do
                                ;                                24-35 - Correct end address
                                ;
                                ;    Entry type 2 (SSCONT) - Bits 3-11 - Unused
                                ;                                12-23 - Number of single steps to do
                                ;                                24-35 - Correct end address
                                ;
                                ;    Entry type 3 (SSCALL) - Bits 3-35 - Address of routine to call
                                ;
                                ;    Entry type 4 (SSCHK) -  Bits 3-35 - Address of routine to call (return
                                ;                                        +1 if done with test, +2 otherwise)
                                ;
                                ;    Entry type 5 (SSJRST) - Bits 3-35 - Table address to continue
                                ;#*******************************************************************

044416  261 17 0 00 000000      SEXEC:  RPUT    (0,1,2)                 ; save AC's

                                ; Check if end of table

044421  200 02 0 06 000000              MOVE    2,(6)                   ; get entry
044422  350 00 0 00 000006              AOS     6                       ; point to next entry
044423  322 02 0 00 044427              JUMPE   2,SEXEX1                ; end of table - yes - exit

                                ; Determine type of entry and dispatch

044424  135 01 0 00 062471              LDB     1,[POINT 3,2,2]         ; get entry
                                        XCT     [JRST   SEXEX1          ; dispatch on it
                                                 JRST   SEXSSS
                                                 JRST   SEXSSC
                                                 JRST   SEXCAL
                                                 JRST   SEXCHK
                                                 JRST   SEXJRS
                                                 JRST   SEXERR
                                                 JRST   SEXERR
                                                 JRST   SEXERR
044425  256 00 0 01 062472                       JRST   SEXERR](1)

                                ; Error in Single Step Table

044426  037 00 0 00 062504      SEXER   [illegible]CD  <Single Step Dispatch Table Error - Entry out of range>
044427  332 00 0 00 064641'     SEXEX1  SKIPE   SDEBUG                  ; EXEC debug mode?
044430  037 00 0 00 030242              PCRL                            ; yes - a final CRLF
044431  262 17 0 00 000002              RGET    (2,1,0)                 ; restore AC's
044434  263 17 0 00 000000              RTN                             ; return
```

```

                                   ; Execute single step

044435  135 00 0 00 062520         SEXSSS: LDB     [POINT 12,2,23]         ; get start address
044436  202 00 0 00 042750*                MOVEM   SNEXT                   ; save it
044437  200 00 0 00 044436*        SEXSSC: MOVE    SNEXT                   ; get next address
044440  202 00 0 00 044551                 MOVEM   SSADDR                  ; save as start address
044441  135 00 0 00 062521                 LDB     [POINT 9,2,11]          ; get # of single steps to do
044442  202 00 0 00 044550                 MOVEM   NSSTEP                  ; save it
044443  135 00 0 00 062522                 LDB     [POINT 12,2,35]         ; get correct end address
044444  202 00 0 00 044552                 MOVEM   SEADDR                  ; save it
044445  350 00 0 00 047012                 AOS     TSTSUB                  ; increment test segment

                                   ; Set up start data

044446  336 01 0 00 042751*                SKIPN   1,SDATA                 ; any data given?
044447  201 01 0 00 000014                 MOVEI   1,MPRUN!4               ; no - set initial data
044450  332 00 0 00 030037                 SKIPE   USER                    ; user mode?
044451  620 01 0 00 000007                 TRZ     1,7                     ; yes - clear PI level bits
044452  660 01 0 00 060010                 TRO     1,SELLAR!MPRUN!SINCYC   ; ensure SELLAR,MPRUN,SINCYC are set
044453  202 01 0 00 044446*                MOVEM   1,SDATA                 ; save start data
044454  260 17 0 00 000000*                GO      SETLAR                  ; set up to write to LAR
044455  255 00 0 00 000000                  JFCL                           ; error

                                   ; Single step NSSTEP times

044456  200 02 0 00 044550                 MOVE    2,NSSTEP                ; get number of times

                                   ; First write the LAR

044457  200 01 0 00 044437*        SEXSS0: MOVE    1,SNEXT                 ; get start address
044460  242 01 0 00 000001                 LSH     1,1                     ; position properly
044461  260 17 0 00 043273*                GO      LDRAR                   ; load the register

                                   ; Then do the single step

044462  200 01 0 00 044453*                MOVE    1,SDATA                 ; get start data
044463  260 17 0 00 042337*                GO      LDCSR                   ; write to CSR

                                   ; Then read the LAR

044464  260 17 0 00 042340*                GO      RDLAR                   ; read LAR
044465  242 01 0 00 777777                 LSH     1,-1                    ; position properly
044466  202 01 0 00 044457*                MOVEM   1,SNEXT                 ; set up new next address
044467  367 02 0 00 044457                 SOJG    2,SEXSS0                ; loop till done
044470  202 01 0 00 044553                 MOVEM   1,SAADDR                ; save actual end address
```

```
                                                ; Then read the LAR

045045  200 01 0 00 045041*                             MOVE    1,SDATA                 ; get start data
045046  620 01 0 00 200010                              TRZ     1,TSTEBF!MPRUN          ; ensure TSTEBF and MPRUN cleared
045047  660 01 0 00 040000                              TRO     1,SELLAR                ;   and set SELLAR
045050  260 17 0 00 045042*                             GO      LDCSR                   ; write to CSR
045051  260 17 0 00 044675*                             GO      RDLAR                   ; read LAR
045052  242 01 0 00 777777                              LSH     1,-1                    ; position properly
045053  202 01 0 00 045036*                             MOVEM   1,SNEXT                 ; set up new next address
045054  367 02 0 00 045036                              SOJG    2,BEXSS0                ; loop till done
045055  202 01 0 00 044553                              MOVEM   1,SAADDR                ; save actual end address

                                                ; Debug output

045056  332 00 0 00 064641'                             SKIPE   SDEBUG                  ; EXEC debug mode?
                                                        GO      [TMSGC  <SSTEP:  #S >   ; yes - print debug
                                                                 MOVE   NSSTEP          ;   output and then
                                                                 PNTDEC                 ;   continue
                                                                 TMSG   <, SADR >
                                                                 MOVE   SSADDR
                                                                 PNTOCS
                                                                 TMSG   <, EADR >
                                                                 MOVE   SEADDR
                                                                 PNTOCS
                                                                 TMSG   <, AADR >
                                                                 MOVE   SAADDR
                                                                 PNTOCS
                                                                 TMSG   <, EBUF >
                                                                 MOVE   AEBUF
                                                                 PNTHW
045057  260 17 0 00 062740                                       RTN]

                                                ; Done - compare end address to expected

045060  402 00 0 00 045143                              SETZM   BFLAG                   ; clear 'incorrect address' flag
045061  200 01 0 00 044553                              MOVE    1,SAADDR                ; save actual end address
045062  312 01 0 00 044552                              CAME    1,SEADDR                ; correct result?
045063  476 00 0 00 045143                              SETOM   BFLAG                   ; no - set 'incorrect address' flag
045064  200 01 0 00 045141                              MOVE    1,AEBUF                 ; get actual EBUF data
045065  316 01 0 00 045142                              CAMN    1,CEBUF                 ; correct result?
045066  350 00 0 17 777774                              AOS     -4(P)                   ; yes - increment return
045067  350 00 0 17 777774                              AOS     -4(P)                   ; set up return
045070  350 00 0 17 777774                              AOS     -4(P)                   ; set up return
045071  262 17 0 00 000003                              RGET    (3,2,1,0)               ; restore AC's
045075  263 17 0 00 000000                              RTN                             ; return +3/+4
```

```
                                                ; Execute a JRST and exit
```

```


                                   ; Execute a JRST and exit

045076  550 02 0 00 000002         BEXJRS: HRRZ    2,2                     ; only want the address
045077  332 00 0 00 064641'                SKIPE   SDEBUG                  ; EXEC debug mode?
                                           GO      [TMSGC  <SJRST Adr  >   ; yes - print debug
                                                    MOVE   2               ;   data and continue
                                                    PNTOCS
045100  260 17 0 00 062554                          RTN]
045101  200 06 0 00 000002                 MOVE    6,2                     ; get new table address
045102  262 17 0 00 000003                 RGET    (3,2,1,0)               ; restore AC's
045106  350 00 0 17 000000                 AOS     (P)                     ; set up return
045107  263 17 0 00 000000                 RTN                             ; return +2

                                   ; Execute a CALL and exit

045110  550 02 0 00 000002         BEXCAL: HRRZ    2,2                     ; only want the address
045111  332 00 0 00 064641'                SKIPE   SDEBUG                  ; EXEC debug mode?
                                           GO      [TMSGC  <SCALL Adr  >   ; yes - print debug
                                                    MOVE   2               ;   data and continue
                                                    PNTOCS
045112  260 17 0 00 062563                          RTN]
045113  261 17 0 00 000004                 PUT     4                       ; save another AC
045114  260 17 1 00 000002                 GO      @2                      ; call the routine
045115  262 17 0 00 000004                 RGET    (4,3,2,1,0)             ; restore AC's
045122  350 00 0 17 000000                 AOS     (P)                     ; set up return
045123  263 17 0 00 000000                 RTN                             ; return +2

                                   ; Execute a CALL and exit (if the return is +1 then exit test)

045124  550 02 0 00 000002         BEXCHK: HRRZ    2,2                     ; only want the address
045125  332 00 0 00 064641'                SKIPE   SDEBUG                  ; EXEC debug mode?
                                           GO      [TMSGC  <SCHK Adr  >    ; yes - print debug
                                                    MOVE   2               ;   data and continue
                                                    PNTOCS
045126  260 17 0 00 062572                          RTN]
045127  261 17 0 00 000004                 PUT     4                       ; save another AC
045130  260 17 1 00 000002                 GO      @2                      ; call the routine
045131  334 00 0 00 000000                 SKIPA                           ; exit test
045132  350 00 0 17 777773                 AOS     -5(P)                   ; ok - set up return
045133  262 17 0 00 000004                 RGET    (4,3,2,1,0)             ; restore AC's
045140  263 17 0 00 000000                 RTN                             ; return +2 or +1

                                   ; Miscellaneous

045141  000000  000000             AEBUF:  0                               ; actual EBUF data
045142  000000  000000             CEBUF:  0                               ; correct EBUF data
045143  000000  000000             BFLAG:  0                               ; 'incorrect address' flag
```

```
                                        ; Determine type of entry and dispatch

045520  135 01 0 00 063013                      LDB     1,[POINT 4,2,3]         ; get entry
                                                XCT     [JRST   CEXERR          ; dispatch on it
                                                          JRST   CEXSRD
                                                          JRST   CEXSWR
                                                          JRST   CEXSRW
                                                          JRST   CEXSTR
                                                          JRST   CEXCAL
                                                          JRST   CEXCAC
                                                          JRST   CEXWAI
                                                          JRST   CEXCOM
                                                          JRST   CEXJRS
                                                          JRST   CEXEXI
                                                          JRST   CEXERR
045521  256 00 0 01 063161                                JRST   CEXERR](1)

                                        ; Error in Test Execute Table

045522  037 00 0 00 063176              CEXERR: TMSGCD  <Data Test Execute Dispatch Table Error - Entry out of range>
045523  332 00 0 00 064641'             CEXEX1: SKIPE   SDEBUG                  ; EXEC debug mode?
045524  037 00 0 00 030242                      PCRL                            ; yes - a final CRLF
045525  262 17 0 00 000002                      RGET    (2,1,0)                 ; restore AC's
045530  263 17 0 00 000000                      RTN                             ; return

                                        ; Set up a read data transfer

045531  550 00 0 00 000002              CEXSRD: HRRZ    2                       ; get transfer length
045532  332 00 0 00 000000                      SKIPE                           ; zero?
045533  202 00 0 00 046036                      MOVEM   CLEN                    ; no - save it
045534  135 00 0 00 063105                      LDB     [POINT 14,2,17]         ; get data pattern
045535  332 00 0 00 000000                      SKIPE                           ; zero?
045536  202 00 0 00 046035                      MOVEM   CPAT                    ; no - save it
045537  201 00 0 00 000001                      MOVEI   1                       ; get type of transfer (read)
045540  202 00 0 00 046037                      MOVEM   CTRAN                   ; save it
045541  201 01 0 00 005000              CSETCC: MOVEI   1,5000                  ; get start address
045542  242 01 0 00 000001                      LSH     1,1                     ; position correctly
045543  260 17 0 00 045263*                     GO      LDRAR                   ; load the register
045544  201 01 0 00 000010                      MOVEI   1,MPRUN                 ; get 'MPRUN' bit
045545  260 17 0 00 045412*                     GO      LDCSR                   ; start up the port
045546  200 01 0 00 063213                      MOVE    1,[400000,,1]           ; get fill pattern
045547  202 01 0 00 050000                      MOVEM   1,BUFF                  ; put fill pattern throughout
045550  200 01 0 00 063214                      MOVE    1,[BUFF,,BUFF+1]        ;   entire buffer area
045551  251 01 0 00 050777                      BLT     1,BUFF+^D511            ;   ...
045552  200 14 0 00 046035                      MOVE    PAT,CPAT                ; get data pattern
045553  200 01 0 00 046037                      MOVE    1,CTRAN                 ; get transfer type
045554  200 02 0 00 046036                      MOVE    2,CLEN                  ; get transfer length
045555  306 01 0 00 000003                      CAIN    1,3                     ; read/write?
045556  242 02 0 00 777777                      LSH     2,-1                    ; yes - divide by 2
045557  260 17 0 00 042521*                     GO      BUFGEN                  ; generate buffer contents
```

```
045560  200 01 0 00 061774                      MOVE    1,[CBUFSZ,,CBUF]        ; get pointer to control buffer
045561  260 17 0 00 042523*                     GO      CHINIT                  ; initialize software
```

```
045560  200 01 0 00 061774                 MOVE    1,[CBUFSZ,,CBUF]          ; get pointer to control buffer
045561  260 17 0 00 042523*                GO      CHINIT                    ; initialize software
045562  551 01 0 00 050000                 HRRZI   1,BUFF                    ; buffer address
045563  200 02 0 00 046036                 MOVE    2,CLEN                    ; word count
045564  200 03 0 00 046037                 MOVE    3,CTRAN                   ; get transfer type
045565  306 03 0 00 000003                 CAIN    3,3                       ; read/write?
045566  242 02 0 00 777777                 LSH     2,-1                      ; yes - divide by 2
045567  400 03 0 00 000000                 SETZ    3,                        ; clear zero fill specifier
045570  260 17 0 00 042527*                GO      GENCCW                    ; generate a CCW list
045571  332 00 0 00 064641'                SKIPE   SDEBUG                    ; EXEC debug mode?
                                           GO      [TMSGC  <CSET>            ; yes - print debug
                                                    MOVE   1,CTRAN           ;   data and continue
                                                    XCT    [TMSG   <RD>
                                                            TMSG   <WR>
                                                            TMSG   <RW>](1)
                                                    TMSG   <  Pat=>
                                                    MOVE   CPAT
                                                    PNTOCS
                                                    TMSG   <  Len=>
                                                    MOVE   CLEN
                                                    PNTOCS
045572  260 17 0 00 063231                          RTN]
045573  350 00 0 17 777775                 AOS     -3(P)                     ; set up proper return
045574  262 17 0 00 000002                 RGET    (2,1,0)                   ; restore AC's
045577  263 17 0 00 000000                 RTN                               ; return

                                   ; Set up a write data transfer

045600  550 00 0 00 000002         CEXSWR: HRRZ    2                         ; get transfer length
045601  332 00 0 00 000000                 SKIPE                             ; zero?
045602  202 00 0 00 046036                 MOVEM   CLEN                      ; no - save it
045603  135 00 0 00 063105                 LDB     [POINT 14,2,17]           ; get data pattern
045604  332 00 0 00 000000                 SKIPE                             ; zero?
045605  202 00 0 00 046035                 MOVEM   CPAT                      ; no - save it
045606  201 00 0 00 000002                 MOVEI   2                         ; get type of transfer (write)
045607  202 00 0 00 046037                 MOVEM   CTRAN                     ; save it
045610  254 00 0 00 045541                 JRST    CSETCC                    ; common code

                                   ; Set up a read/write data transfer

045611  550 00 0 00 000002         CEXSRW: HRRZ    2                         ; get transfer length
045612  332 00 0 00 000000                 SKIPE                             ; zero?
045613  202 00 0 00 046036                 MOVEM   CLEN                      ; no - save it
045614  135 00 0 00 063105                 LDB     [POINT 14,2,17]           ; get data pattern
045615  332 00 0 00 000000                 SKIPE                             ; zero?
045616  202 00 0 00 046035                 MOVEM   CPAT                      ; no - save it
045617  201 00 0 00 000003                 MOVEI   3                         ; get type of transfer (read/write)
045620  202 00 0 00 046037                 MOVEM   CTRAN                     ; save it
045621  200 14 0 00 046035                 MOVE    PAT,CPAT                  ; get data pattern
045622  254 00 0 00 045541                 JRST    CSETCC                    ; common code
```

```
046233  201 00 0 00 000077                MOVEI   77                      ; get 'no interrupt' code
046234  137 00 0 00 063410                DPB     [POINT 6,1,8]           ; save it
046235  202 01 0 00 046574                MOVEM   1,IFLAG                 ; save interrupt type
046236  332 00 0 00 064641'               SKIPE   SDEBUG                  ; test debug mode?
                                          GO      [TMSGC  <ISETDE  Iflag >; yes - print debug
                                                   MOVE   IFLAG           ;   data and continue
                                                   PNTHW
                                                   TMSG   < Adr >
                                                   MOVE   1,IIADR
                                                   PNTOCS
                                                   TMSG   < Data >
                                                   MOVE   1,IIDEPC
                                                   PNTOCS
                                                   RTN]
046237  260 17 0 00 063437                AOS     -3(P)                   ; set up proper return
046240  350 00 0 17 777775                RGET    (2,1,0)                 ; restore AC's
046241  262 17 0 00 000002                RTN                             ; return
046244  263 17 0 00 000000

                                ; Set up an inc/dec operation

046245  135 01 0 00 063406      IEXSID: LDB     1,[POINT 18,2,35]       ; get address of list
046246  200 00 0 01 000000                MOVE    (1)                     ; get address
046247  202 00 0 00 046600                MOVEM   IIADR                   ; save it
046250  200 00 0 01 000001                MOVE    1(1)                    ; get data
046251  202 00 0 00 046603                MOVEM   IIEXAC                  ; save it
046252  202 00 0 00 046604                MOVEM   IIEXAA                  ; save it
046253  202 00 0 00 046605                MOVEM   IIDEPC                  ; save it
046254  202 00 0 00 046606                MOVEM   IIDEPA                  ; save it
046255  202 00 0 00 046601                MOVEM   IIINCC                  ; save it
046256  200 00 0 00 063213                MOVE    [400000,,1]             ; get null data
046257  202 00 0 00 046602                MOVEM   IIINCA                  ; save it as actual
046260  200 01 0 00 046600                MOVE    1,IIADR                 ; include address
046261  201 00 0 00 000004                MOVEI   4                       ; get IOP function
046262  137 00 0 00 063407                DPB     [POINT 3,1,2]           ; save it
046263  201 00 0 00 000077                MOVEI   77                      ; get 'no interrupt' code
046264  137 00 0 00 063410                DPB     [POINT 6,1,8]           ; save it
046265  202 01 0 00 046574                MOVEM   1,IFLAG                 ; save interrupt type
046266  332 00 0 00 064641'               SKIPE   SDEBUG                  ; test debug mode?
                                          GO      [TMSGC  <ISETID  Iflag >; yes - print debug
                                                   MOVE   IFLAG           ;   data and continue
                                                   PNTHW
                                                   TMSG   < Adr >
                                                   MOVE   1,IIADR
                                                   PNTOCS
                                                   TMSG   < Data >
                                                   MOVE   1,IIINCC
                                                   PNTOCS
                                                   RTN]
046267  260 17 0 00 063455                AOS     -3(P)                   ; set up proper return
046270  350 00 0 17 777775                RGET    (2,1,0)                 ; restore AC's
046271  262 17 0 00 000002                RTN                             ; return
046274  263 17 0 00 000000
```

```

                                ; Set up an examine/increment operation

```

```


                                     ; Set up an examine/increment operation

 046275  135 01 0 00 063406          IEXSEI: LDB    1,[POINT 18,2,35]        ; get address of list
 046276  200 00 0 01 000000                  MOVE   (1)                      ; get address
 046277  202 00 0 00 046600                  MOVEM  IIADR                    ; save it
 046300  200 00 0 01 000001                  MOVE   1(1)                     ; get data
 046301  350 00 0 00 000000                  AOS                             ; get correct data
 046302  202 00 0 00 046603                  MOVEM  IIEXAC                   ; save it
 046303  202 00 0 00 046605                  MOVEM  IIDEPC                   ; save it
 046304  202 00 0 00 046606                  MOVEM  IIDEPA                   ; save it
 046305  202 00 0 00 046601                  MOVEM  IIINCC                   ; save it
 046306  200 00 0 00 063213                  MOVE   [400000,,1]              ; get null data
 046307  202 00 0 00 046602                  MOVEM  IIINCA                   ; save it as actual
 046310  202 00 0 00 046604                  MOVEM  IIEXAA                   ; save it as actual
 046311  200 01 0 00 046600                  MOVE   1,IIADR                  ; include address
 046312  201 00 0 00 000006                  MOVEI  6                        ; get IOP function
 046313  137 00 0 00 063407                  DPB    [POINT 3,1,2]            ; save it
 046314  201 00 0 00 000077                  MOVEI  77                       ; get 'no interrupt' code
 046315  137 00 0 00 063410                  DPB    [POINT 6,1,8]            ; save it
 046316  202 01 0 00 046574                  MOVEM  1,IFLAG                  ; save interrupt type
 046317  332 00 0 00 064641'                 SKIPE  SDEBUG                   ; test debug mode?
                                             GO     [TMSGC  <ISETEI  Iflag >; yes - print debug
                                                     MOVE   IFLAG            ;   data and continue
                                                     PNTHW
                                                     TMSG   < Adr >
                                                     MOVE   1,IIADR
                                                     PNTOCS
                                                     TMSG   < Data >
                                                     MOVE   1,IIEXAC
                                                     PNTOCS
                                                     RTN]
 046320  260 17 0 00 063473                  AOS    -3(P)                    ; set up proper return
 046321  350 00 0 17 777775                  RGET   (2,1,0)                  ; restore AC's
 046322  262 17 0 00 000002
 046325  263 17 0 00 000000                  RTN                             ; return

                                     ; Set up interrupt (non-vectored or vectored)

 046326  260 17 0 00 042714*          IEXSIN: GO     INITPI                   ; init PI system
 046327  260 17 0 00 042715*                 GO     SETVEC                   ; set up vector addresses
 046330  200 01 0 00 046570                  MOVE   1,IADDR                  ; get address
 046331  135 00 0 00 063105                  LDB    [POINT 14,2,17]          ; get IOP function
 046332  137 00 0 00 063407                  DPB    [POINT 3,1,2]            ; position correctly for IFLAG
 046333  135 00 0 00 063406                  LDB    [POINT 18,2,35]          ; get interrupt type expected
 046334  137 00 0 00 063410                  DPB    [POINT 6,1,8]            ; position correctly for IFLAG
 046335  202 01 0 00 046574                  MOVEM  1,IFLAG                  ; save interrupt type
 046336  200 00 0 00 046601                  MOVE   IIINCC                   ; get correct increment data
 046337  202 00 0 00 046602                  MOVEM  IIINCA                   ; save actual increment data
 046340  200 00 0 00 046603                  MOVE   IIEXAC                   ; get correct examine data
 9669  046341  202 00 0 00 046604          MOVEM   IIEXAA                  ; save actual examine data
 9670  046342  200 00 0 00 046605          MOVE    IIDEPC                  ; get correct deposit data
 9671  046343  202 00 0 00 046606          MOVEM   IIDEPA                  ; save actual deposit data
 9672  046344  332 00 0 00 064641'         SKIPE   SDEBUG                  ; test debug mode?
 9673                                      GO      [TMSGC  <ISETIN  Iflag >; yes - print debug
 9674                                               MOVE   IFLAG            ;   data and continue
 9675                                               PNTHW
 9676  046345  260 17 0 00 063511                   RTN]
 9677  046346  350 00 0 17 777775          AOS     -3(P)                   ; set up proper return
 9678  046347  262 17 0 00 000002          RGET    (2,1,0)                 ; restore AC's
 9679  046352  263 17 0 00 000000          RTN                             ; return
 9680
 9681                                 ; Start the port
 9682
 9683  046353  260 17 0 00 045623*  IEXSTR: GO      IPACLR                  ; do a 'port clear'
 9684  046354  200 00 0 00 063515          MOVE    [INTNUM,,INTNUM+1]      ; build BLT pointer
 9685  046355  251 00 0 00 000000*         BLT     INTEND                  ; clear interrupt history data
 9686  046356  135 01 0 00 063042          LDB     1,[POINT 32,2,35]       ; get start address
 9687  046357  301 01 0 00 007777          CAIL    1,7777                  ; default address selected?
 9688  046360  200 01 0 00 046570          MOVE    1,IADDR                 ; yes - get it
 9689  046361  202 01 0 00 045627*         MOVEM   1,SNEXT                 ; save it
 9690  046362  202 01 0 00 046570          MOVEM   1,IADDR                 ; save as start address
 9691  046363  242 01 0 00 000001          LSH     1,1                     ; position properly
 9692  046364  260 17 0 00 045632*         GO      LDRAR                   ; load the register
 9693  046365  336 01 0 00 045641*         SKIPN   1,SDATA                 ; any data given?
 9694  046366  201 01 0 00 000014          MOVEI   1,MPRUN!4               ; no - set initial data
 9695  046367  332 00 0 00 030037          SKIPE   USER                    ; user mode?
 9696  046370  620 01 0 00 000007          TRZ     1,7                     ; yes - clear PI level bits
 9697  046371  660 01 0 00 040010          TRO     1,SELLAR!MPRUN          ; ensure SELLAR,MPRUN are set
 9698  046372  202 01 0 00 046365*         MOVEM   1,SDATA                 ; save start data
 9699  046373  200 01 0 00 046372*         MOVE    1,SDATA                 ; get start data
 9700  046374  260 17 0 00 045753*         GO      LDCSR                   ; write to CSR
 9701  046375  332 00 0 00 064641'         SKIPE   SDEBUG                  ; EXEC debug mode?
 9702                                      GO      [TMSGC  <ISTART  Adr  > ; yes - print debug
 9703                                               MOVE   IADDR            ;   data and continue
 9704                                               PNTOCS
 9705  046376  260 17 0 00 063522                   RTN]
 9706  046377  350 00 0 17 777775          AOS     -3(P)                   ; set up return
 9707  046400  262 17 0 00 000002          RGET    (2,1,0)                 ; restore AC's
 9708  046403  263 17 0 00 000000          RTN                             ; return +3/+4
```

```

                                        ; Execute a JRST and exit

046404  550 02 0 00 000002      IEXJRS: HRRZ    2,2                             ; only want the address
046405  332 00 0 00 064641'             SKIPE   SDEBUG                          ; EXEC debug mode?
                                        GO      [TMSGC  <IJRST Adr  >           ; yes - print debug
                                                 MOVE   2                       ;   data and continue
                                                 PNTOCS
046406  260 17 0 00 063531                       RTN]
046407  200 06 0 00 000002              MOVE    6,2                             ; get new table address
046410  262 17 0 00 000002              RGET    (2,1,0)                         ; restore AC's
046413  350 00 0 17 000000              AOS     (P)                             ; set up return
046414  263 17 0 00 000000              RTN                                     ; return +2

                                        ; Execute a CALL and exit

046415  550 02 0 00 000002      IEXCAL: HRRZ    2,2                             ; only want the address
046416  332 00 0 00 064641'             SKIPE   SDEBUG                          ; EXEC debug mode?
                                        GO      [TMSGC  <ICALL Adr  >           ; yes - print debug
                                                 MOVE   2                       ;   data and continue
                                                 PNTOCS
046417  260 17 0 00 063540                       RTN]
046420  261 17 0 00 000003              RPUT    (3,4)                           ; save some AC's
046422  260 17 1 00 000002              GO      @2                              ; call the routine
046423  262 17 0 00 000004              RGET    (4,3,2,1,0)                     ; restore AC's
046430  350 00 0 17 000000              AOS     (P)                             ; set up return
046431  263 17 0 00 000000              RTN                                     ; return +2

                                        ; Execute a CALL and check if test done and exit

046432  550 02 0 00 000002      IEXCAC: HRRZ    2,2                             ; only want the address
046433  332 00 0 00 064641'             SKIPE   SDEBUG                          ; EXEC debug mode?
                                        GO      [TMSGC  <ICALLC Adr  >          ; yes - print debug
                                                 MOVE   2                       ;   data and continue
                                                 PNTOCS
046434  260 17 0 00 063547                       RTN]
046435  261 17 0 00 000003              RPUT    (3,4)                           ; save some AC's
046437  260 17 1 00 000002              GO      @2                              ; call the routine
046440  334 00 0 00 000000              SKIPA                                   ; exit test
046441  350 00 0 17 777773              AOS     -5(P)                           ; set up return
046442  262 17 0 00 000004              RGET    (4,3,2,1,0)                     ; restore AC's
046447  263 17 0 00 000000              RTN                                     ; return +2
```

```

                                        ; Exit test (if error flag is set)

046450  332 00 0 00 064641'             IEXEXI: SKIPE   SDEBUG                          ; EXEC debug mode?
                                                GO      [TMSGC  <IEXIT>                 ; yes - print debug
046451  260 17 0 00 063555                               RTN]                           ; data and continue
046452  336 00 0 00 000015                      SKIPN   ERFLG                           ; any errors yet?
046453  350 00 0 17 777775                      AOS     -3(P)                           ; no - not done with test yet
046454  262 17 0 00 000002                      RGET    (2.1,0)                         ; restore AC's
046457  263 17 0 00 000000                      RTN                                     ; return +2 or +1

                                        ; Wait for interrupt operation to complete

046460  350 00 0 00 047012              IEXWAI: AOS     TSTSUB                          ; increment test segment
046461  135 00 0 00 063042                      LDB     [POINT 32,2,35]                 ; get time to wait
046462  260 17 0 00 043121                      GO      ODELAY                          ; wait
046463  260 17 0 00 045733*                     GO      RDCSR                           ; read CSR
046464  254 00 0 00 046543                      JRST    IEXWA5                          ; error accessing CSR
046465  202 01 0 00 046572                      MOVEM   1,ICSR                          ; save CSR data
046466  260 17 0 00 000000*                     GO      .PIOFF                          ; shut off PI system

                                        ; Find out final address

046467  201 01 0 00 040000                      MOVEI   1,SELLAR                        ; gset up to read LAR
046470  260 17 0 00 046374*                     GO      LDCSR                           ; write CSR
046471  260 17 0 00 045754*                     GO      RDLAR                           ; read LAR
046472  242 01 0 00 777777                      LSH     1,-1                            ; position correctly
046473  202 01 0 00 046571                      MOVEM   1,IADDRF                        ; save it

                                        ; Handle inc/dec/exam/depos location

046474  135 02 0 00 063557                      LDB     2,[POINT 3,IFLAG,2]             ; get interrupt type
046475  307 02 0 00 000001                      CAIG    2,1                             ; vectored or non-vectored?
046476  254 00 0 00 046521                      JRST    IEXWA0                          ; yes - continue
046477  306 02 0 00 000002                      CAIN    2,2                             ; no - examine?
                                                GO      [GO     IEXWRG                  ; yes - read examine data
                                                         MOVEM  1,IEBUF                 ; save EBUF data
                                                         MOVEM  1,IIEXAA                ; save for compare
046500  260 17 0 00 063560                               RTN]
046501  306 02 0 00 000006                      CAIN    2,6                             ; examine/increment?
                                                GO      [GO     IEXWRG                  ; yes - read examine data
                                                         MOVEM  1,IEBUF                 ; save EBUF data
                                                         MOVEM  1,IIEXAA                ; save for compare
046502  260 17 0 00 063560                               RTN]
046503  200 00 0 00 046604                      MOVE    IIEXAA                          ; get actual data examined
046504  312 00 0 00 046603                      CAME    IIEXAC                          ; same as correct data?
046505  474 15 0 00 000000                      SETO    ERFLG,                          ; no - set error flag
046506  336 00 0 00 046600                      SKIPN   IIADR                           ; valid address of data?
046507  254 00 0 00 046521                      JRST    IEXWA0                          ; no - skip over these compares
```

```
046510  200 01 1 00 046600                  MOVE    1,@IIADR                        ; get actual data
046511  312 01 0 00 046601                  CAME    1,IIINCC                        ; same as correct?
046512  474 15 0 00 000000                  SETO    ERFLG,                          ; no - set error flag
046513  312 01 0 00 046605                  CAME    1,IIDEPC                        ; same as correct?
046514  474 15 0 00 000000                  SETO    ERFLG,                          ; no - set error flag
046515  301 02 0 00 000004                  CAIL    2,4                             ; increment of some sort?
046516  202 01 0 00 046602                  MOVEM   1,IIINCA                        ; yes - save actual data
046517  306 02 0 00 000003                  CAIN    2,3                             ; deposit?
046520  202 01 0 00 046605                  MOVEM   1,IIDEPC                        ; yes - save actual data

                                    ; Find out interrupt data

046521  335 00 0 00 042747*         IEXWA0: SKIPGE  INTNUM                          ; any interrupts occurred?
                                            JRST    [SETOM   IITYPE                 ; no - flag such and clear
                                                     SETOM   IINTPC                 ;   interrupt PC and
046522  254 00 0 00 063564                           JRST    IEXWA1]                ;   continue
046523  200 01 0 00 046521*                 MOVE    1,INTNUM                        ; get interrupt number
046524  301 01 0 00 000024                  CAIL    1,^D20                          ; overflow interrupt?
046525  201 01 0 00 000023                  MOVEI   1,^D19                          ; yes - get last saved interrupt
046526  550 00 0 01 042706*                 HRRZ    INTTYP(1)                       ; get interrupt type
046527  202 00 0 00 046576                  MOVEM   IITYPE                          ; save it
046530  200 00 0 01 042663*                 MOVE    INTPC(1)                        ; get interrupt PC
046531  202 00 0 00 046577                  MOVEM   IINTPC                          ; save it

                                    ; Now determine if an error occurred due to receipt or lack of receipt of
                                    ; an interrupt.

046532  135 02 0 00 063567          IEXWA1: LDB     2,[POINT 6,IFLAG,8]             ; get interrupt type
046533  302 02 0 00 000077                  CAIE    2,77                            ; interrupt expected?
046534  254 00 0 00 046540                  JRST    IEXWA2                          ; yes - continue
046535  331 00 0 00 046523*                 SKIPL   INTNUM                          ; interrupt occurred?
046536  474 15 0 00 000000                  SETO    ERFLG,                          ; yes - set error flag
046537  254 00 0 00 046544                  JRST    IEXWA6                          ; continue

                                    ; Interrupt was expected - verify that the correct one really did occur

046540  312 02 0 01 046526*         IEXWA2: CAME    2,INTTYP(1)                     ; correct type interrupt occurred?
046541  474 15 0 00 000000                  SETO    ERFLG,                          ; no - set error flag
046542  254 00 0 00 046544                  JRST    IEXWA6                          ; continue

                                    ; Error did occur/exit

046543  474 15 0 00 000000          IEXWA5: SETO    ERFLG,                          ; set error flag
046544  332 00 0 00 064641'         IEXWA6: SKIPE   SDEBUG                          ; EXEC debug mode?
                                            GO      [TMSGC   <IWAIT CSR >           ; yes - print debug
                                                     MOVE    ICSR                   ;   data and continue
                                                     PNTHW
                                                     TMSG    < Stopped at >
                                                     MOVE    IADDRF
                                                     PNTOCS
046545  260 17 0 00 063576                           RTN]
```

```
DFPTA
DFPTA

REL
RELIA
REQ1
REQ2
RESCR
RESCR
RESQA
RFROM
RLOCE
RLOCE
RSTAR
RTN
RTO
RUNTM
S1
S2
SAADD
SAVAD
SAVCM
SAVCR
SAVCR
SAVCR
SAVFL
SAVHS
SAVMA
SAVRA
SB
SCOON
SCOPE
SCOSW
SDATA
SDEBU
SEADD
SELLA
SELTS
SELTS
SEN1T
SENT1
SENT1
SEQ
SETEE
SETLA
SETVD
SEXCA
SEXCM
SEXEC
SEXEF
SEXEX
SEXJM
SEXSS
SEXSS
SEXSS
SHWC
SINCV
SLAST
```

```
  9851  046546  260 17 0 00 000000*              GO      .PION                    ; turn interrupt system back on
  9852  046547  336 00 0 00 000015               SKIPN   ERFLG                    ; error flag set?
  9853  046550  350 00 0 17 777775               AOS     -3(P)                    ; no - set up good return
  9854  046551  350 00 0 17 777775               AOS     -3(P)                    ; set up return
  9855  046552  350 00 0 17 777775               AOS     -3(P)                    ; set up return
  9856  046553  262 17 0 00 000002               RGET    (2,1,0)                  ; restore AC's
  9857  046556  263 17 0 00 000000               RTN                              ; return
  9858
  9859                                   ; Routine to read 2901 register with the examine data it it
  9860
  9861  046557  201 01 0 00 001100       IEXWRG: MOVEI   1,1100                   ; yes -  get start address
  9862  046560  242 01 0 00 000001               LSH     1,1                      ; position correctly
  9863  046561  260 17 0 00 046364*              GO      LDRAR                    ; load the register
  9864  046562  201 01 0 00 020010               MOVEI   1,SINCYC!MPRUN           ; get proper bits
  9865  046563  260 17 0 00 046470*              GO      LDCSR                    ; write CSR
  9866  046564  201 01 0 00 200000               MOVEI   1,TSTEBF                 ; set up to read EBUF
  9867  046565  260 17 0 00 046563*              GO      LDCSR                    ; write CSR
  9868  046566  260 17 0 00 045740*              GO      RDEBUF                   ; read the EBUF
  9869  046567  263 17 0 00 000000               RTN                              ; return
  9870
  9871                                   ; Miscellaneous
  9872
  9873  046570  000000  000000           IADDR:  0                        ; initial starting address
  9874  046571  000000  000000           IADDRF: 0                        ; final address
  9875  046572  000000  000000           ICSR:   0                        ; final CSR data
  9876  046573  000000  000000           ITIME:  0                        ; amount of time to let port run
  9877  046574  000000  000000           IFLAG:  0                        ; interrupt should occur data
  9878  046575  000000  000000           IIOPF:  0                        ; IOP function word
  9879  046576  000000  000000           IITYPE: 0                        ; type of interrupt which occurred
  9880  046577  000000  000000           IINTPC: 0                        ; interrupt PC
  9881  046600  000000  000000           IIADR:  0                        ; address of location to be inc/dec/exa/dep
  9882  046601  000000  000000           IIINCC: 0                        ; contents of location to be inc/dec (correct)
  9883  046602  000000  000000           IIINCA: 0                        ; contents of location to be inc/dec (actual)
  9884  046603  000000  000000           IIEXAC: 0                        ; data examined (correct)
  9885  046604  000000  000000           IIEXAA: 0                        ; data examined (actual)
  9886  046605  000000  000000           IIDEPC: 0                        ; data deposited (correct)
  9887  046606  000000  000000           IIDEPA: 0                        ; data deposited (actual)
  9888  046607  000000  000000           IEBUF:  0                        ; EBUF contents (of 2901 register 5)
```

```
TSTC5
TSTC5
TSTC6
TSTC6
TSTC7
TSTCL
TSTCN
TSTDC
TSTD1
TSTD1
TSTD1
TSTD1
TSTD2
TSTD3
TSTD4
TSTD5
TSTD6
TSTD7
TSTD8
TSTD9
TSTDD
TSTDE
TSTDE
TSTDE
TSTDE
TSTDE
TSTDE
TSTDI
TSTDI
TSTDL
TSTDS
TSTDS
TSTDS
TSTE1
TSTE1
TSTE1
TSTE1
TSTE1
TSTE1
TSTE1
TSTE1
TSTE1
TSTE2
TSTE2
TSTE2
TSTE2
TSTE2
TSTE2
TSTE2
TSTE2
TSTE3
TSTE3
TSTE3
TSTE3
```

```

                                 ;#*********************************************************************
                                 ;* IIPNT - Print out error data
                                 ;#*********************************************************************

046610  261 17 0 00 000000       IIPNT:  RPUT    (0,1)                   ; save AC's
046612  200 01 0 00 046103*              MOVE    1,$COSW                 ; get switches

                                 ; Print start/end address

046613  603 01 0 00 000200               TLNE    1,TXTINH                ; full printout?
                                         GO      [TMSGC  <Start >        ; no
                                                  MOVE   IADDR
                                                  PNT4
                                                  TMSG   <  End >        ; print end address
                                                  MOVE   IADDRF
                                                  PNT4
046614  260 17 0 00 063605                        RTN]
046615  603 01 0 00 000200               TLNE    1,TXTINH                ; full printout
046616  254 00 0 00 046633               JRST    IIPNO                   ; no - continue
046617  037 00 0 00 062576               TMSGC   <Start Addr:  >         ; print start address
046620  200 00 0 00 046570               MOVE    IADDR
046621  037 04 0 00 000000               PNT4
046622  037 00 0 00 063337               TMSGC   <End Addr:    >         ; print end address
046623  200 00 0 00 046571               MOVE    IADDRF
046624  037 04 0 00 000000               PNT4
046625  037 00 0 00 063155               TMSGC   <Final CSR data:  >
046626  200 00 0 00 046572               MOVE    ICSR
046627  037 13 0 00 000000               PNTHW                           ; print it
046630  037 00 0 00 063614               TMSGC   <Final EBUF data:  >
046631  200 00 0 00 046607               MOVE    IEBUF
046632  037 13 0 00 000000               PNTHW                           ; print it

                                 ; Print IOP function word

046633  037 00 0 00 063621       IIPNO:  TMSGC   <IOP Function word:  >
046634  200 00 0 00 046575               MOVE    IIOPF
046635  037 13 0 00 000000               PNTHW
046636  603 01 0 00 000200               TLNE    1,TXTINH                ; full printout?
046637  254 00 0 00 046705               JRST    IIPN3                   ; no - continue
046640  037 00 0 00 063626               TMSG    <, AdrSpa = >
046641  135 01 0 00 063631               LDB     1,[POINT 3,IIOPF,2]     ; get address space
                                         XCT     [TMSG   <EPT>
                                                  TMSG   <ExecVir>
                                                  TMSG   <Illegal Function>
                                                  TMSG   <Illegal Function>
046642  256 00 0 01 063643                        TMSG   <Physical>](1)  ; print it
```

```
DFPTA
DFPTA

TSTU2
TSTU2
TSTU2
TSTU2
TSTU2
TSTU2
TSTU3
TSTU3
TSTU3
TSTU3
TSTU3
TSTU3
TSTU3
TSTU3
TSTU3
TSTU4
TSTU4
TSTU4
TSTU5
TSTU6
TSTU7
TTALT
TTIDE
TTIME
TTNBR
TTPNO
TTPN1
TTPN2
TTPNT
TTPNX
TTSIX
TTYYE
TUSER
TWORD
TXALL
TXTIN
UDEBU
UNAME
USER
USRIO
UUT
WRDER
XWORD
XWORD
XXW
XXX
ZAL
ZCB
ZEB
ZMP
ZSE
$$TOG
$ARG2
$B
$CHR
```

AADDR

```
046643  037 00 0 00 063650                 TMSG     <, Func = >
046644  135 01 0 00 063652                 LDB      1,[POINT 3,IIOPF,5]     ; get function
                                           XCT      [TMSG   <Non-Vect Int,> ; print it
                                                     TMSG   <Non-Vect Int,>
                                                     TMSG   <Vect Int>
                                                     GO     [MOVE    IIOPF
                                                             TLNN    4000
                                                             TMSG    <Increment,>
                                                             TLNE    4000
                                                             TMSG    <Decrement,>
                                                             RTN]
                                                     TMSG   <Examine,>
                                                     TMSG   <Deposit,>
                                                     TMSG   <Byte,>
046645  256 00 0 01 063705                           TMSG   <Examine/Incr,>](1)
046646  037 00 0 00 063715                 TMSGC    <   Qual = >
046647  135 00 0 00 063720                 LDB      [POINT 1,IIOPF,6]       ; get QUAL bit
046650  037 16 0 00 000003                 PNTOCS                           ; print it
046651  037 00 0 00 063721                 TMSG     <, DevNum = >
046652  135 00 0 00 063724                 LDB      [POINT 4,IIOPF,10]      ; get device number
046653  037 16 0 00 000003                 PNTOCS                           ; print it
046654  037 00 0 00 063725                 TMSG     <, Addr = >
046655  135 00 0 00 063727                 LDB      [POINT 24,IIOPF,35]     ; get address
046656  037 16 0 00 000003                 PNTOCS                           ; print it

                                   ; Print type of interrupt expected

046657  135 01 0 00 063567         IIPN1:  LDB      1,[POINT 6,IFLAG,8]     ; get interrupt expected
046660  306 01 0 00 000077                 CAIN     1,77                    ; any interrupt expected?
046661  037 00 0 00 063730                 TMSGC    <No interrupt expected> ; no - print such
046662  306 01 0 00 000077                 CAIN     1,77                    ; any interrupt expected?
046663  254 00 0 00 046673                 JRST     IIPN2                   ; no - continue
046664  037 00 0 00 063735                 TMSGC    <Interrupt function attempted:  >
046665  135 01 0 00 063557                 LDB      1,[POINT 3,IFLAG,2]     ; get type
                                           XCT      [TMSG   <Non-Vectored interrupt>
                                                     TMSG   <Vectored interrupt>
                                                     TMSG   <Examine addr >
                                                     TMSG   <Deposit addr >
                                                     TMSG   <Increment addr >
                                                     TMSG   <Decrement addr >
046666  256 00 0 01 064000                           TMSG   <Examine/increment addr >](1)
046667  307 01 0 00 000001                 CAIG     1,1                     ; vectored or non-vectored?
046670  254 00 0 00 046673                 JRST     IIPN2                   ; yes - continue
046671  135 00 0 00 064007                 LDB      [POINT 18,IFLAG,35]     ; get address
046672  037 16 0 00 000003                 PNTOCS                           ; print it out
```

```

                                    ; Print which type of interrupt did occur

046673  335 01 0 00 046576          IIPN2:  SKIPGE  1,IITYPE                ; any interrupt occurred?
046674  037 00 0 00 064010                  TMSGC   <No interrupt occurred> ; no - print such
046675  321 01 0 00 046705                  JUMPL   1,IIPN3                 ; any interrupt occurred? no - continue
046676  336 00 0 00 000001                  SKIPN   1                       ; vectored interrupt occurred?
046677  037 00 0 00 064015                  TMSGC   <Vectored interrupt occurred>
046700  332 00 0 00 000001                  SKIPE   1                       ; non-vectored interrupt occurred?
046701  037 00 0 00 064023                  TMSGC   <Non-Vectored interrupt occurred>
046702  037 00 0 00 064032                  TMSG    < from PC >
046703  200 00 0 00 046577                  MOVE    IINTPC                  ; get PC
046704  037 16 0 00 000003                  PNTOCS                          ; print it

                                    ; Print increment/decrement c/a

046705  200 01 0 00 046612*         IIPN3:  MOVE    1,SCOSW                 ; get switches
046706  200 00 0 00 046601                  MOVE    IIINCC                  ; get correct inc/dec data
046707  316 00 0 00 046602                  CAMN    IIINCA                  ; same as actual?
046710  254 00 0 00 046730                  JRST    IIPN4                   ; yes - continue
046711  037 00 0 00 064034                  TMSGC   <Address inc/dec:  >
046712  200 00 0 00 046600                  MOVE    IIADR                   ; get address inc/dec
046713  037 16 0 00 000003                  PNTOCS                          ; print it
046714  603 01 0 00 000200                  TLNE    1,TXTINH                ; full printout
046715  037 00 0 00 064041                  TMSGC   < (C):  >               ; no
046716  607 01 0 00 000200                  TLNN    1,TXTINH                ; full printout
046717  037 00 0 00 064043                  TMSGC   <  Correct:  >          ; yes
046720  200 00 0 00 046601                  MOVE    IIINCC
046721  037 13 0 00 000000                  PNTHW
046722  603 01 0 00 000200                  TLNE    1,TXTINH                ; full printout
046723  037 00 0 00 064046                  TMSGC   < (A):  >               ; no
046724  607 01 0 00 000200                  TLNN    1,TXTINH                ; full printout
046725  037 00 0 00 064050                  TMSGC   <   Actual:  >          ; yes
046726  200 00 0 00 046602                  MOVE    IIINCA
046727  037 13 0 00 000000                  PNTHW

                                    ; Print examine c/a

046730  200 01 0 00 046705*         IIPN4:  MOVE    1,SCOSW                 ; get switches
046731  200 00 0 00 046603                  MOVE    IIEXAC                  ; get correct examine data
046732  316 00 0 00 046604                  CAMN    IIEXAA                  ; same as actual?
046733  254 00 0 00 046753                  JRST    IIPN5                   ; yes - continue
046734  037 00 0 00 064053                  TMSGC   <Address examined:  >
046735  200 00 0 00 046600                  MOVE    IIADR                   ; get address inc/dec
046736  037 16 0 00 000003                  PNTOCS                          ; print it
046737  603 01 0 00 000200                  TLNE    1,TXTINH                ; full printout
046740  037 00 0 00 064041                  TMSGC   < (C):  >               ; no
046741  607 01 0 00 000200                  TLNN    1,TXTINH                ; full printout
046742  037 00 0 00 064043                  TMSGC   <  Correct:  >          ; yes
046743  200 00 0 00 046603                  MOVE    IIEXAC
046744  037 13 0 00 000000                  PNTHW
```

```
BEXSSC
BEXSSS
BFLAG
BONETM
BUFCOM
BUFF
BUFGEN
CADDR


CALL
CALMAR
CALPAR
CBASE
CBUF
CBUFSZ
CBUS
CCMIC
CCPNO
CCPN2
CCPN3
CCPN4
CCPNT
CCPNX
CCSR
CCTAB1
CCTAB3
CCWPNT
CDERR
CEBUAD
CEBUF
CEBUFA
CEBUFC
CEXCAC
CEXCAL
CEXCOM
CEXEC
CEXERR
CEXEX1
CEXEXI
CEXJRS
CEXSRD
CEXSRW
CEXSTR
CEXSWF
CEXWA2
CEXWA3
CEXWA4
CEXWAI
CHDATA
CHINIT
CHKARG
```

```
 10032  046745  603 01 0 00 000200               TLNE    1,TXTINH                ; full printout
 10033  046746  037 00 0 00 064046               TMSGC   < (A):  >               ; no
 10034  046747  607 01 0 00 000200               TLNN    1,TXTINH                ; full printout
 10035  046750  037 00 0 00 064050               TMSGC   <   Actual:  >          ; yes
 10036  046751  200 00 0 00 046604               MOVE    IIEXAA
 10037  046752  037 13 0 00 000000               PNTHW
 10038
 10039                                   ; Print deposit c/a
 10040
 10041  046753  200 01 0 00 046730*      IIPN5:  MOVE    1,SCOSW                 ; get switches
 10042  046754  200 00 0 00 046605               MOVE    IIDEPC                  ; get correct deposit data
 10043  046755  316 00 0 00 046606               CAMN    IIDEPA                  ; same as actual?
 10044  046756  254 00 0 00 046776               JRST    IIPNX                   ; yes - continue
 10045  046757  037 00 0 00 064060               TMSGC   <Address deposited:  >
 10046  046760  200 00 0 00 046600               MOVE    IIADR                   ; get address inc/dec
 10047  046761  037 16 0 00 000003               PNTOCS                          ; print it
 10048  046762  603 01 0 00 000200               TLNE    1,TXTINH                ; full printout
 10049  046763  037 00 0 00 064041               TMSGC   < (C):  >               ; no
 10050  046764  607 01 0 00 000200               TLNN    1,TXTINH                ; full printout
 10051  046765  037 00 0 00 064043               TMSGC   <  Correct:  >          ; yes
 10052  046766  200 00 0 00 046605               MOVE    IIDEPC
 10053  046767  037 13 0 00 000000               PNTHW
 10054  046770  603 01 0 00 000200               TLNE    1,TXTINH                ; full printout
 10055  046771  037 00 0 00 064046               TMSGC   < (A):  >               ; no
 10056  046772  607 01 0 00 000200               TLNN    1,TXTINH                ; full printout
 10057  046773  037 00 0 00 064050               TMSGC   <   Actual:  >          ; yes
 10058  046774  200 00 0 00 046606               MOVE    IIDEPA
 10059  046775  037 13 0 00 000000               PNTHW
 10060
 10061                                   ; Exit
 10062
 10063  046776  262 17 0 00 000001       IIPNX:  RGET    (1,0)                   ; restore AC's
 10064  047000  263 17 0 00 000000               RTN                             ; return
 10065
```

```
CHKCSR
CHKTS0
CHKTS1
CHKTST
CHKTSX
CIPORT
CISEL

CLEN
CLOGBF
CLRBUF
CLREBU
CMD0
CMD1
CMD2
CMD3
CMD3A
CMD4
CMDQUE
CMDQUU
CONFIG
CONPNT
CONSW
CONVSX
CPAT
CRAMFR

CRAMPE
CRAMTC
CSERR
CSETCC
CSHFLG
CSHMEM
CSRENC
CSREN1
CSREN2
CSRENE
CSRENG
CSRENS
CSRENX
CSRPNT
CSRPNX
CSTATF
CSTWRD
CTIME
CTRAN
CURTIM
CWORDL
```

```
                                                  SUBTTL  Miscellaneous Test Variables

                                          ;#*******************************************************************
                                          ; Port device code selection
                                          ;
                                          ; Select word (NISEL or CISEL):
                                          ;  Bit 0 - Port exists - 0 - no   (This is determined by whether or not
                                          ;                        1 - yes   a CONI succeeds and bit 0 is set,
                                          ;                                   indicating a port).
                                          ;      1 - Port selectable - 0 - no   (This is determined by whether
                                          ;                            1 - yes   or not a DEVREQ succeeds).
                                          ;      2 - Port selected - 0 - no
                                          ;                          1 - yes
                                          ;#*******************************************************************

 047001  056400  000000                   PORTNI: 56400,,0                        ; NI port device code
 047002  057400  000000                   PORTCI: 57400,,0                        ; CI port device code
 047003  000000  000000                   NISEL:  0                               ; NI port selected
 047004  000000  000000                   CISEL:  0                               ; CI port selected
 047005  000000  000000                   PORSEL: 0                               ; port currently selected
 047006  000000  000000                   UUT:    0                               ; unit under test


                                          ;#*******************************************************************
                                          ; Current Test Data
                                          ;#*******************************************************************

 047007  000 00 0 00 000000               TSTNAM: Z                               ; current test description
 047010  000 00 0 00 000000               TSTCLS: Z                               ; current test class (0-5)
 047011  000 00 0 00 000000               TSTNUM: Z                               ; current test number (1-n)
 047012  000 00 0 00 000000               TSTSUB: Z                               ; current subtest number
 047013  000 00 0 00 000000               TSTPC:  Z                               ; current test PC
 047014  000 00 0 00 000000               TSTFLG: Z                               ; current test flags
 047015  000 00 0 00 000000               TSTADD: Z                               ; current test dispatch table address
 047016  000 00 0 00 000000               TSTMIC: Z                               ; current test microcode address
 047017  000 00 0 00 000000               TSTMSK: Z                               ; mask to determine what test to run
 047020  000 00 0 00 000000               TSTREP: Z                               ; number of test repetitions done
 047021  000 00 0 00 000000               TSTREQ: Z                               ; number of test repetitions requested
 047022  000 00 0 00 000000               TSTMUL: Z                               ; multiple test flag (0-no,1-yes)
 047023  000 00 0 00 000000               TSTINH: Z                               ; inhibit fault isolation flag
 047024  000 00 0 00 000000               TSTHIG: Z                               ; addr of high probability modules
 047025  000 00 0 00 000000               TSTLOW: Z                               ; addr of low probability modules
 047026  000 00 0 00 000000               TSTOFF: Z                               ; relocatable offset of test module
 047027  000 00 0 00 000000               TSTREL: Z                               ; reliability mode repeat count
 047030  000 00 0 00 000000               TSTCNT: Z                               ; reliability mode repeat count
 047031  000 00 0 00 000000               FLTCLS: Z                               ; failing test class
 047032  000 00 0 00 000000               FLTNUM: Z                               ; failing test number
 047033  000 00 0 00 000000               NERROR: Z                               ; number of errors so far

 047034  000 00 0 00 000000               PMODE:  Z                               ; program mode
```

```
CWORDR



D1LIT
DACRAM
DBCRAM
DBPNT
DBPNT0
DBPNT1
DBREA0
DBREAK
DC1ST
DCBR0
DCBR1
DCBRK
DCCOD1
DCCOD2
DCCOD3
DCCOD4
DCCOD5
DCCOD6
DCCODE
DCLEAR
DCMAR0
DCMAR1
DCMAR2
DCMARK
DCODPB
DCOLDB
DCOM0
DCOM1
DCOM3A
DCOMA0
DCOMA1
DCOMA2
DCOMA3
DCOMA4
DCOMA5
DCOMAA
DCOMAG
DCOMAL
DCOMAX
DCOMDB
DCOMDE
DCOMEX
DCOMEZ
DCOMGO
DCOML0
DCOMLI
DCOMLX
DCOMPA
```

```

                                  ;#**********************************************************************
                                  ;* Data Buffer Areas For Reading And Writing
                                  ;#**********************************************************************

050000                            LOC     <.+1000>&777000

050000                            BUFF:   BLOCK   ^D1024


                                  ;#*********************************************************************
                                  ;* Data transfer paramters
                                  ;#*********************************************************************

052000  000000  000100            TWORD:  100                             ; number of words
052001  000000  000000            TMODE:  0                               ; data format
052002  222333  444555            TPAT:   222333,,444555                  ; selected data pattern

052003                            CBUF:   BLOCK   CBUFSZ                  ; channel control word storage area

052320                            CSTWRD: BLOCK   4                       ; channel logout storage


                                  ;#*********************************************************************
                                  ; PATCH - Patch Area
                                  ;#*********************************************************************

052324                            PATCH0:
052324                            PATCH:  BLOCK   100                     ; patch area
```

```
DCOMSG
DCOMZE
DCONI
DCONI1
DCONO
DCONO1
DCONT
DCWPNT
DDALU
DDALU0
DDALUM
DDALUX
DDATI
DDATI1
DDATO
DDATO1
DDCRAM
DDCSR
DDEBU1
DDEBUF
DDELA0
DDELAX
DDELAY
DDLOC0
DDLOCM
DDLOCS
DDLOCX
DDNEX0
DDNEX1
DDNEX2
DDNEXT
DDRAR
DDTAB
DEALU
DEALU0
DEALU1
DEALUM
DEALUX
DEBTIM
DEBUG1
DEBUG2
DEBUG3
DEBUG4
DEBUG5
DECRAM
DECSR
DECVER
DECYN
DEEBUF
DELAR
DELOC0
DELOC1
DELOCM
```

```
                                     ;#*****************************************************************
                                     ;* Special code to test SPEAR entries
                                     ;#*****************************************************************

052424  200 00 0 00 055325           STEST:  MOVE    [JFCL]                  ; get noop
052425  202 00 0 00 044213                   MOVEM   S1                      ; delete SYERR call #1
052426  202 00 0 00 044327                   MOVEM   S2                      ; delete SYERR call #2
052427  260 17 0 00 052433                   GO      .+4                     ; do it
052430  201 01 0 00 044220                   MOVEI   1,SENT10                ; get address to print
052431  260 17 0 00 052466                   GO      STESTP                  ; print it
052432  254 00 0 00 052436                   JRST    STEST0                  ; continue
052433  261 17 0 00 000000                   RPUT    (0,1)                   ; save AC's
052435  254 00 0 00 044205                   JRST    SPCHK1                  ; check entry type 1

052436  260 17 0 00 052442           STEST0: GO      .+4                     ; do it
052437  201 01 0 00 044342                   MOVEI   1,SENT11                ; get address to print
052440  260 17 0 00 052466                   GO      STESTP                  ; print it
052441  254 00 0 00 052451                   JRST    STEST1                  ; continue
052442  261 17 0 00 000000                   RPUT    (0,1,2,3,4)             ; save AC's
052447  400 15 0 00 000000                   SETZ    ERFLG,                  ; clear error flag
052450  254 00 0 00 044256                   JRST    SPCHK2                  ; check entry type 2 (no error)

052451  260 17 0 00 052455           STEST1: GO      .+4                     ; do it
052452  201 01 0 00 044342                   MOVEI   1,SENT11                ; get address to print
052453  260 17 0 00 052466                   GO      STESTP                  ; print it
052454  254 00 0 00 030007                   JRST    SRTDDT                  ; continue
052455  261 17 0 00 000000                   RPUT    (0,1,2,3,4)             ; save AC's
052462  474 15 0 00 000000                   SETO    ERFLG,                  ; set error flag
052463  201 00 0 00 000012                   MOVEI   12                      ; get test number
052464  202 00 0 00 047011                   MOVEM   TSTNUM                  ; save it
052465  254 00 0 00 044256                   JRST    SPCHK2                  ; check entry type 2 (error)

052466  037 00 0 00 064065           STESTP: TMSG    <Buffer:>
052467  505 01 0 00 777753                   HRLI    1,-25                   ; build AOBJN word
052470  200 00 0 01 000000                   MOVE                            ; get data
052471  037 13 0 00 000000                   PNTHW                           ; print
052472  037 00 0 00 030242                   PCRL
052473  253 01 0 00 052470                   AOBJN   1,STESTP+2              ; loop till done
052474  263 17 0 00 000000                   RTN                             ; return
```

```
                                        ;#**********************************************************************
                                        ;* Special code to produce an EXE version
                                        ;#**********************************************************************

052475  200 00 0 00 064070              EXEN:   MOVE    [SKIPA]
052476  202 00 0 00 000000*                     MOVEM   REQ1
052477  202 00 0 00 000000*                     MOVEM   REL
052500  200 00 0 00 064071              EXE:    MOVE    [10,,0]
052501  202 00 0 00 030046                      MOVEM   CONSW
052502  200 00 0 00 064072                      MOVE    [SETOM $$TOGG]
052503  202 00 0 00 030641                      MOVEM   START+31
052504  201 00 0 00 030000                      MOVEI   30000
052505  542 00 0 00 000120                      HRRM    .JBSA
052506  402 00 0 00 030056                      SETZM   $ONETM
052507  402 00 0 00 000000*                     SETZM   HAVECI
052510  402 00 0 00 000000*                     SETZM   HAVENI
052511  402 00 0 00 000016                      SETZM   MBCN
052512  402 00 0 00 064646'                     SETZM   TSLOD1
052513  402 00 0 00 064647'                     SETZM   TSLOD2
052514  254 00 0 00 030007                      JRST    SRTDDT


                                        ;#**********************************************************************
                                        ; End of Mainline Module
                                        ;#**********************************************************************

052515                                  D1LIT:  XLIST

NO ERRORS DETECTED

PROGRAM BREAK IS 064661
ABSOLUTE BREAK IS 064567
CPU TIME USED 01:12.825

143P CORE USED
```

```
DISQUU
DLBR0
DLBR1
DLBR2
DLBRK
DLBRX
DLCRAM
DLGPNT
DLIST
DLMAR0
DLMAR1
DLMAR2
DLMARK
DLMARX
DLOAD
DMARK
DMARK0
DRBR0
DRBRK
DRCRAM

DRDINC
DRDINI
DRESET
DRMAR0
DRMARK
DROUT1
DROUTN
DSADDR
DSDATA
DSET
DSETA1
DSETA2
DSETAD
DSETB1
DSETC1
DSETC2
DSETCS
DSETD1
DSETD2
DSETD3
DSETDA
DSETDX
DSETE1
DSETE2
DSETEB
DSETF0
DSETF1
DSETF2
DSETFA
DSETFN
DSETHE
DSETHL
```

0417

```
DFPTA1  Test Control and Port Debugger  MACRO %53B(1242) 07:46  1-Jul-85 Page S-1                                              SEQ 0430
DFPTA1  MAC      23-May-85 20:41        SYMBOL TABLE

AADDR           000000  ext   CALPAR          043625  ent   CMD3A           034440        DCMARK          041326
AAPNT           044732  ent   CBASE           030651  ext   CMD4            034442        DCODPB          040221
AAPNT0          044752        CBUF            052003  ent   CMDQUE          034457        DCOLDB          040177
AAPNT1          044763        CBUFSZ          000315  spd   CMDQUU          034462        DCOM0           037726
AAPNTX          044765        CBUS    100000  000000  spd   CONFIG          032314        DCOM1           037743
ABORT           400000  spd   CCMIC           042432        CONPNT          032400        DCOM3A          040122
ADRPE   100000  000000  spd   CCPN0           046064        CONSW           030046        DCOMA0          040071
AEBUF           045141  int   CCPN2           046103        CONVSX          040125  ext   DCOMA1          040074
AEXEC           044617  ent   CCPN3           046111        CPAT            046035  int   DCOMA2          040117
AEXEC0          044667        CCPN4           046135        CRAMFR          037746        DCOMA3          040120
AEXECX          044702        CCPNT           046044  ent   CRAMPE  004000  000000  spd   DCOMA4          040137
ALACT           044725  int   CCPNX           046143        CRAMTO          037747        DCOMA5          040141
ALCOR           044724  int   CCSR            046027  int   CSERR           046033        DCOMAA          040107
ALCSR           044721  int   CCTAB1          042372        CSETCC          045541        DCOMAG          040055
ALEBF           044722  int   CCTAB3          042412        CSHFLG          030506        DCOMAL          040052
ALERR           044723  int   CCWPNT          042575  ext   CSHMEM          030507        DCOMAX          040144
ALFLC           044730  int   CDERR           046034  int   CSREN0          043751        DCOMDB          040045
ALFLE           044727  int   CEBUAD          064573'       CSREN1          043765        DCOMDE          037756
ALFLS           044726  int   CEBUF           045142  int   CSREN2          043766        DCOMEX          040325
ALFLT           044731  int   CEBUFA          046032  int   CSRENB          043776        DCOMEZ          040346
ALL     770000  000000  spd   CEBUFC          046031  int   CSRENG          043737  ent   DCOMGO          037705
ALSRT           044720  int   CEXCAC          045700        CSRENS          064574'       DCOMLO          040353
ALTCHK  007000  000000        CEXCAL          045663        CSRENX          043770        DCOMLI          040351
ALTF            064570' int   CEXCOM          046004        CSRPNT          043715  ent   DCOMLX          040373
ALTIM           044717  int   CEXEC           045512  ent   CSRPNX          043732        DCOMPA          042610
ALTMGO          030063        CEXERR          045522        CSTATF          046040  int   DCOMSG          040155
ALU     020000  000000  spd   CEXEX1          045523        CSTWRD          052320  ent   DCOMZE          040310
ALUFR           040753        CEXEXI          045716        CTIME           046030        DCONI           036016
ALUTO           040754        CEXJRS          045652        CTRAN           046037        DCONI1          036021
ANEXT           037755        CEXSRD          045531        CURTIM          033027  ext   DCONO           036026
ARGFLG          041667  ext   CEXSRW          045611        CWORDL          037751  int   DCONO1          036032
ARGUM           054544  ext   CEXSTR          045623        CWORDR          037752  int   DCONT           037551
BBPN0           045170        CEXSWR          045600        D1LIT           052515        DCWPNT          042572
BBPN1           045177        CEXWA2          045747        DACRAM          037674        DDALU           040557
BBPN2           045216        CEXWA3          045771        DBCRAM          037671        DDALU0          040563
BBPNT           045144  ent   CEXWA4          045772        DBPNT           042537        DDALUM          040735
BBPNX           045224        CEXWAI          045726        DBPNT0          042552        DDALUX          040635
BEGIN           030000        CHDATA          045762  ext   DBPNT1          042564        DDATI           036037
BEXCAL          045110        CHINIT          045561  ext   DBREA0          041636        DDATI1          036042
BEXCHK          045124        CHKARG          055564  ext   DBREAK          041634        DDATO           036046
BEXEC           044770  ent   CHKCSR          000000  ext   DC1ST           064575'       DDATO1          036052
BEXERR          045002        CHKTS0          043054        DCBR0           041731        DDCRAM          037666
BEXEX1          045003        CHKTS1          043071        DCBR1           041742        DDCSR           036001
BEXJRS          045076        CHKTST          043043        DCBRK           041712        DDEBU1          037613
BEXSSO          045036        CHKTSX          043072        DCCOD1          042322        DDEBUF          037607
BEXSSC          045014        CIPORT          000020  spd   DCCOD2          042346        DDELA0          043166
BEXSSS          045012        CISEL           047004  ent   DCCOD3          042350        DDELAX          043202
BFLAG           045143        CLEN            046036  int   DCCOD4          042356        DDELAY          043156
BONETM          064571'       CLOGBF          046041        DCCOD5          042360        DDLOC0          041064
BUFCOM          046142  ext   CLRBUF          060315  ext   DCCOD6          042366        DDLOCM          041230
BUFF            050000  int   CLREBU          043240  ext   DCCODE          042314        DDLOCS          041060
BUFGEN          045557  ext   CMD0            034371        DCLEAR          036113        DDLOCX          041131
CADDR           037750  int   CMD1            034376        DCMAR0          041345        DDNEX0          041553
CALL    200000  000000  spd   CMD2            034403        DCMAR1          041356        DDNEX1          041562
CALMAR          064572'       CMD3            034427        DCMAR2          041373        DDNEX2          041567
```

0418

```
DFPTA1  Test Control and Port Debugger  MACRO %53B(1242) 07:46  1-Jul-85 Page S-2                                              SEQ 0431
```

0418

```
DFPTA1  Test Control and Port Debugger  MACRO %53B(1242) 07:46  1-Jul-85 Page S-2
DFPTA1  MAC      23-May-85 20:41        SYMBOL TABLE                                  SEQ 0431

DDNEXT          041532          DFVERX          042167          DSETC1          035474          DZALUM          040750
DDRAR           037641          DFVERY          042222          DSETC2          035475          DZALUX          040727
DDTAB           041537          DHELP           035315          DSETCS          035465          DZCRAM          037677
DEALU           040451          DHELP0          035327          DSETD1          035607          DZCSR           036010
DEALU0          040472          DHELP1          035360          DSETD2          035610          DZEBUF          037602
DEALU1          040512          DIINIT          042711          DSETD3          035620          DZLOC0          041155
DEALUM          040732          DIPRIN          042635          DSETDA          035562          DZLOC1          041175
DEALUX          040552          DISGO           031374          DSETDX          035603          DZLOCM          041243
DEBTIM          043170  ext     DISLIS          031304          DSETE1          035533          DZLOCS          041134
DEBUG1          002000  spd     DISMS   104000  000167  int     DSETE2          035534          DZLOCX          041220
DEBUG2          001000  spd     DISP0           030735          DSETEB          035522          DZRAR           037653
DEBUG3          000400  spd     DISP0A          031013          DSETF0          035636          EADDR           000000  ext
DEBUG4          000200  spd     DISP0B          031015          DSETF1          035641          EBUS    400000  000000  spd
DEBUG5          000100  spd     DISP1           031023          DSETF2          035642          ENATST          064600'
DECRAM          037663          DISP2           031067          DSETFA          035666          ENDFIX          030577
DECSR           035771          DISPAT          030731          DSETFN          035622          ENDUU0  037500  000004
DECVER          000004  spd     DISPD           031051          DSETHE          035753          EOPUU0  037700  000004
DECYN           035526  ext     DISPE           031064          DSETHL          035761          ERESET          036111  ext
DEEBUF          037566          DISPR           031036          DSETL1          035554          ERFLG           000015
DELAR           037621          DISPX           031070          DSETL2          035555          ERRCNT          064601'
DELOC0          040776          DISQUE          031074          DSETLN          035542          ERRPC           030052
DELOC1          041017          DISQUU          031077          DSETP1          035513          EXE             052500
DELOCM          041223          DLBR0           041775          DSETP2          035514          EXEN            052475
DELOCS          040755          DLBR1           042010          DSETPL          035716          FGETW           042263  ext
DELOCX          041053          DLBR2           042027          DSETPP          035735          FINCMD          060265  ext
DENE0           041464          DLBRK           041756          DSETPR          035502          FINECH          064602' int
DENEX0          041516          DLBRX           042035          DSETPX          035751          FINIT           042261  ext
DENEX1          041522          DLCRAM          037702          DSETUP          032321  ext     FINPUT          064603' int
DENEX2          041526          DLGPNT          042600          DSETWH          035652          FIOFF           062050  ext
DENEXT          041455          DLIST           042062          DSINIT          036120          FLDLIS          040243
DESTA0          041610          DLMAR0          041416          DSPAT           035767          FLTCH0          034272
DESTAK          041574          DLMAR1          041431          DSPEAR          004000  spd     FLTCH1          034304
DESTAX          041625          DLMAR2          041445          DSSCLR          036274          FLTCHK          034267
DESTAY          041626          DLMARK          041377          DSSPN0          036263          FLTCHX          034313
DESTKM          041631          DLMARX          041453          DSSPNT          036252          FLTCLS          047031
DETAB           041502          DLCAD           042037          DSST0           036217          FLTHA0          034253
DEVREL          032306  ext     DMARK           041250          DSST1           036233          FLTHAN          034251
DEVREQ          034374  ext     DMARK0          041252          DSSTEP          036203          FLTNUM          047032
DFEXA0          042263          DRBR0           041671          DSSTX           036237          FLTPNT          064604'
DFEXAM          042256          DRBRK           041667          DSTAR0          036137          FMTTYP          064605'
DFEXAX          042302          DRCRAM          040430  ent     DSTART          036133          FOARG           000000  ext
DFILLN          042443          DRDINC          042513          DSTOP           036160          FORPNT          033321  ext
DFILLP          042466          DRDINI          042507          DSTOP1          036173          FSELEC  037240  000004
DFILN0          042454          DRESET          036106          DSTPAT          035770          GENCCW          045570  ext
DFILP0          042477          DRMAR0          041301          DSWIT           042624          GET     262740  000000
DFLAG           064576'         DRMARK          041277          DSWORD          035766          GETJI   104000  000507  int
DFLIS0          042235          DROUT1          036063          DTRAC0          036205          GETLOG          042604  ext
DFLIST          042227          DROUTN          036057          DTRACE          036200          GETPPN          044404
DFLISX          042254          DSADDR          035762          DTRANS          042304          GO      260740  000000
DFLOA0          042106          DSDATA          035763          DTRFLG          064577'         GTAD    104000  000227  int
DFLOAD          042077          DSET            035403          DWCRAM          040376  ent     HALT    254200  000000
DFLOAX          042121          DSETA1          035457          DWRINI          042532          HAVECI          052507  ext
DFVER0          042145          DSETA2          035460          DZALU           040640          HAVENI          052510  ext
DFVERF          042135          DSETAD          035444          DZALU0          040661          HIBER   047000  000072
DFVERR          042177          DSETB1          035700          DZALU1          040701          IADDR           046570  int
```

0419

```
DFPTA1  Test Control and Port Debugger  MACRO %53B(1242) 07:46  1-Jul-85 Page S-3
```

0419

```
DFPTA1  Test Control and Port Debugger  MACRO %53B(1242) 07:46  1-Jul-85 Page S-3
DFPTA1  MAC     23-May-85 20:41         SYMBOL TABLE                                             SEQ 0432

IADDRF          046571        INTUSE          042671  ext   MLOAD0          043243        PATPNT          035742  ext
ICSR            046572  int   IPACLR          046353  ext   MLOAD1          043322        PCMD            064625'
IDEBUG          064606' int   IPAER    000200 000000  spd   MLOADN          043207  ent   PCRL     037000 030242
IEBUF           046607        IPASRT          037554  ext   MLOADX          043325        PCRL2    037000 030243
IEXCAC          046432        IPASST          041616  ext   MLOADY          043211  ent   PCRLF    037040 030242
IEXCAL          046415        IPASTP          036163  ext   MMPROC          000002  spd   PFORCE   037040 000026
IEXEC           046146  ent   IPRIN0          042643        MONTYP          030516        PFSET           030654  ext
IEXERR          046156        IPRINI          042654        MPER     200000 000000  spd   PGMINT   265000 030011
IEXEX1          046157        IPRINT          042640  ent   MPROC    200000 000000  spd   PGMNAM          030600  int
IEXEXI          046450        IPRINX          042650        MPRUN           000010  spd   PJRST    254000 000000
IEXJRS          046404        ISOPN0          031535        MSG      100000 000000  spd   PMODE           047034  int
IEXSDE          046215        ISOPN1          031546        MULFLG          064616'       PNT1     037040 000000
IEXSEI          046275        ISOPN2          031550        MULINI          064617'       PNT4     037200 000000
IEXSEX          046165        ISOPN3          031554        MVCOM           043224        PNT4F    037200 000001
IEXSID          046245        ISOPN4          031571        MVFLAG          064620'       PNTAL    037740 000000
IEXSIN          046326        ISOPN5          031575        MVNUM           064621'       PNTCIF   037040 000000
IEXSTR          046353        ISOPN6          031613        MVPNT           043357        PNTCRM          043507
IEXWA0          046521        ISOPN7          031627        MVPNT1          043377        PNTDCF   037640 000001
IEXWA1          046532        ISOPN8          031633        MVPNTW          043436        PNTDEC   037640 000000
IEXWA2          046540        ISOPNT          031515        MVPNTX          043426        PNTDEV          042726  ext
IEXWA5          046543        ISOPNX          031634        MVREP           064622'       PNTENB          030220
IEXWA6          046544        ISTOP           043237  ext   MWBADR          043340        PNTHST          037504
IEXWAI          046460        ITERAT          000005  spd   MWBLAC          043346        PNTHW    037540 000000
IEXWRG          046557        ITIME           046573        MWBLCO          043343        PNTMSF   037040 000000
IFLAG           046574  int   ITRCNT          030024        MWBRAC          043354        PNTMSG   037000 000000
IIADR           046600        LARG2           064607'       MWBRCO          043351        PNTOCC   001000 000000
IIDEPA          046606        LASARG          035652  ext   MWDQUE          040266        PNTOCF   037740 000003
IIDEPC          046605        LASDTC          042531        NCBUS    100000 000000  spd   PNTOCS   037700 000003
IIEXAA          046604        LAST     010000 000000  spd   NDCB     000200 000000  spd   PNTSIX   037000 000002
IIEXAC          046603        LAXER    000400 000000  spd   NDMP     000400 000000  spd   PNTSXF   037040 000002
IIINCA          046602        LDCRAM          043276  ext   NEBUS    400000 000000  spd   PNTWD           043603
IIINCC          046601        LDCSR           046565  ext   NERROR          047033        PORSEL          047005  ent
IINTPC          046577        LDEBUF          037617  ext   NETBAD          031753        PORTCI          047002  ent
IIOPF           046575  int   LDEBUG          064610' int   NETDES          032063        PORTNI          047001  ent
IIPN0           046633        LDRAR           046561  ext   NETPNT          031643        PRSFLG          035765
IIPN1           046657        LGWC     000100 000000  spd   NEXM     020000 000000  spd   PSP      037000 000040
IIPN2           046673        LOCFR           041246        NIPORT          000040  spd   PSPF     037040 000040
IIPN3           046705        LOCTO           041247        NISEL           047003  ent   PTEST           032520
IIPN4           046730        LOGBUF          042603  ext   NMBUS    040000 000000  spd   PTEST0          032535
IIPN5           046753        LOGPNT          046107  ext   NMPROC   200000 000000  spd   PTESTP          032537
IIPNT           046610  ent   LOOPGM          040000  spd   NOTWC0   040000 000000  spd   PTESTX          032550
IIPNX           046776        LOOPTS          020000  spd   NSSTEP          044550  int   PUT      261740 000000
IITYPE          046576        MAPNEW          030143        ODELA0          043131        PVPAGI          030511
INDLIS          031275  ent   MBARG           064611'       ODELA1          043140        QADDR           046025  int
INHFLT          200000  spd   MBARGF          064612'       ODELAY          043121  ent   QADDRF          046026  int
INITPD          032277  ext   MBCN            000016        OPRSEL          000010  spd   RALUB           064626'
INITPI          046326  ext   MBUS     010000 000000  spd   OVN      000020 000000  spd   RALUE           064627'
INTAPR          042700  ext   MCBUS           000001  spd   P               000017        RANCNT          064630'
INTCON          042675  ext   MCNVER          000000  spd   PARFLG          035764  int   RANDBS          030022
INTCSR          042703  ext   MCPUS           064613'       PASCNT          030047        RANGEN          032775
INTEND          055553  ext   MDEBUG          064614' int   PA[illegible]   000014        RANNUM          064631'
INTNUM          055554  ext   MLIST           043446        PA[illegible]   064623'       RDCRAM          062202  ext
INTPC           046530  ext   MLIST0          043455        PA[illegible]   064624'       RDCSR           046463  ext
INTTIM          042666  ext   MLISTX          043500        PATCH           052324        RDEBUF          046566  ext
INTTYP          046540  ext   MLNUM           064615'       PATCH0          052324        RDLAR           046471  ext
```

0420

```
DFPTA1  Test Control and Port Debugger  MACRO %53B(1242) 07:46  1-Jul-85 Page S-4
```

```
DFPTA1  Test Control and Port Debugger  MACRO %53B(1242) 07:46  1-Jul-85 Page S-4
DFPTA1  MAC      23-May-85 20:41         SYMBOL TABLE                                                     SEQ 0433

REL             052477  ext   SLDATA          064642'       TEND            033701        TSTA36          000000  ext
RELIAB          000400  spd   SNEXT           063051  ext   TEXCAC          045330        TSTA37          000000  ext
REQ1            052476  ext   SPCHK1          044205        TEXCAL          045313        TSTA4           000000  ext
REQ2            000000  ext   SPCHK2          044256        TEXCH0          045411        TSTA40          000000  ext
RESCR0          044073        SPEAR1          064643'       TEXCHK          045356        TSTA41          000000  ext
RESCRM          044062        SPEAR2          064644' ent   TEXC00          045260        TSTA42          000000  ext
RESQAV          000200  spd   SPRE1X          044215        TEXCON          045257        TSTA5           000000  ext
RFROM           037754        SPRE2A          044247        TEXEC           045227  ent   TSTA6           000000  ext
RLOCB           064632'       SPRE2B          044307        TEXERR          045237        TSTA7           000000  ext
RLOCE           064633'       SPRE2C          044320        TEXEX1          045240        TSTADD          047015  ent
RSTART          200000  spd   SPRE2X          044334        TEXEXI          045346        TSTAL0          033274
RTN      263740 000000        SPREP1          044171        TEXJRS          045302        TSTALT          033265
RTO             037753        SPREP2          044234  ent   TEXSST          045246        TSTC1           000000  ext
RUNTME          043167  ext   SRTDDT          030007        TEXSTR          045253        TSTC10          000000  ext
S1              044213        SRTLIS          000026  spd   TFAIL           045437        TSTC11          000000  ext
S2              044327        SSADDR          044551  int   TLERR           043713        TSTC12          000000  ext
SAADDR          044553  int   SSADRL          036504        TLOAD           043661  ent   TSTC13          000000  ext
SAVADD          064634'       SSADRN          036704        TLOADE          043703        TSTC14          000000  ext
SAVCMD          064635'       SSCSRL          037104        TLOADX          043701        TSTC15          000000  ext
SAVCR0          044047        SSCSRN          037304        TMODE           052001        TSTC16          000000  ext
SAVCRL          044107        SSLOC           036303        TMP1            064645'       TSTC17          000000  ext
SAVCRM          044040        SSNUM           036304        TMPROC          033560        TSTC2           000000  ext
SAVFLD          064636'       SSPN1           044577        TPAT            052002  int   TSTC20          000000  ext
SAVHST          037523        SSPNT           044554  ent   TRACE           042717  ent   TSTC21          000000  ext
SAVMAX          064637'       SSPNX           044614        TRACE0          042743        TSTC22          000000  ext
SAVRAR          064640'       START           030610  ent   TRACET          400000  spd   TSTC23          000000  ext
SB              030607        STARTA          030710  ent   TRACEX          042771        TSTC24          000000  ext
SCOOFF          042746  ext   STARTB          030727        TSEQ            033426        TSTC25          000000  ext
SCOPER   027000 000000        STCLOK          030652  ext   TSLOD1          064646' int   TSTC26          000000  ext
SCOSW           046753  ext   STEST           052424        TSLOD2          064647' int   TSTC27          000000  ext
SDATA           046373  ext   STEST0          052436        TSTA1           000000  ext   TSTC3           000000  ext
SDEBUG          064641' int   STEST1          052451        TSTA10          000000  ext   TSTC30          000000  ext
SEADDR          044552  int   STESTP          052466        TSTA11          000000  ext   TSTC31          000000  ext
SELLAR          040000  spd   STRT1           030644        TSTA12          000000  ext   TSTC32          000000  ext
SELTST          043023        SWCHPT          000000  ext   TSTA13          000000  ext   TSTC33          000000  ext
SELTSX          043037        SWCOM           042632  ext   TSTA14          000000  ext   TSTC34          000000  ext
SEN11L          044355        SWITT           044253  ext   TSTA15          000000  ext   TSTC35          000000  ext
SENT10          044220        SWPTAB          030525        TSTA16          000000  ext   TSTC36          000000  ext
SENT11          044342        SWRGT           030633  ext   TSTA17          000000  ext   TSTC37          000000  ext
SEQ      040000 000000  spd   SWSTP           000116  spd   TSTA2           000000  ext   TSTC4           000000  ext
SETEBU          037614  ext   SWSTT           000036  spd   TSTA20          000000  ext   TSTC40          000000  ext
SETLAR          044673  ext   SYERR    104000 000527  int   TSTA21          000000  ext   TSTC41          000000  ext
SETVEC          046327  ext   TADDR           045430  int   TSTA22          000000  ext   TSTC42          000000  ext
SEXCAL          044515        TADDRF          045431  int   TSTA23          000000  ext   TSTC43          000000  ext
SEXCHK          044532        TAKFIL          032747  int   TSTA24          000000  ext   TSTC44          000000  ext
SEXEC           044416  ent   TALU            033516        TSTA25          000000  ext   TSTC45          000000  ext
SEXERR          044426        TCBUS           033621        TSTA26          000000  ext   TSTC46          000000  ext
SEXEX1          044427        TCOMP           045436        TSTA27          000000  ext   TSTC47          000000  ext
SEXJRS          044504        TCSRF           045432  int   TSTA3           000000  ext   TSTC5           000000  ext
SEXSS0          044457        TDENA    000100 000000  spd   TSTA30          000000  ext   TSTC50          000000  ext
SEXSSC          044437        TDISP1          033330        TSTA31          000000  ext   TSTC51          000000  ext
SEXSSS          044435        TDISP2          033702        TSTA32          000000  ext   TSTC52          000000  ext
SHWC     000040 000000  spd   TEBUFA          045435  int   TSTA33          000000  ext   TSTC53          000000  ext
SINCYC          020000  spd   TEBUFC          045434  int   TSTA34          000000  ext   TSTC54          000000  ext
SLAST           037535  ext   TEBUS           033330        TSTA35          000000  ext   TSTC55          000000  ext
```

```
DFPTA1  Test Control and Port Debugger  MACRO %53B(1242) 07:46  1-Jul-85 Page S-5
```

```
TSTC56          000000  ext     TSTE33          000000  ext     TSTL1           064650'         TSTS40          000000  ext
TSTC57          000000  ext     TSTE34          000000  ext     TSTL2           064651'         TSTS41          000000  ext
TSTC6           000000  ext     TSTE35          000000  ext     TSTL3           064652'         TSTS42          000000  ext
TSTC60          000000  ext     TSTE36          000000  ext     TSTLI0          032173          TSTS43          000000  ext
TSTC7           000000  ext     TSTE37          000000  ext     TSTLI1          032210          TSTS44          000000  ext
TSTCLS          047010  ent     TSTE4           000000  ext     TSTLIS          032166          TSTS45          000000  ext
TSTCNT          047030          TSTE40          000000  ext     TSTLOW          047025          TSTS46          000000  ext
TSTD0           033065          TSTE41          000000  ext     TSTLSS          032250          TSTS47          000000  ext
TSTD1           033107          TSTE42          000000  ext     TSTLST          032216          TSTS5           000000  ext
TSTD10          033231          TSTE43          000000  ext     TSTMBO          031476          TSTS50          000000  ext
TSTD11          033246          TSTE44          000000  ext     TSTMBU          031463          TSTS51          000000  ext
TSTD12          033255          TSTE45          000000  ext     TSTMIC          047016  ent     TSTS52          000000  ext
TSTD2           033121          TSTE46          000000  ext     TSTMSK          047017  ent     TSTS53          000000  ext
TSTD3           033122          TSTE47          000000  ext     TSTMUL          047022          TSTS54          000000  ext
TSTD4           033125          TSTE5           000000  ext     TSTNAM          047007  ent     TSTS55          000000  ext
TSTD5           033130          TSTE50          000000  ext     TSTNUM          047011  ent     TSTS56          000000  ext
TSTD6           033153          TSTE51          000000  ext     TSTOFF          047026  ent     TSTS57          000000  ext
TSTD7           033165          TSTE52          000000  ext     TSTPC           047013  ent     TSTS6           000000  ext
TSTD8           033207          TSTE53          000000  ext     TSTPNO          043111          TSTS60          000000  ext
TSTD9           033214          TSTE54          000000  ext     TSTPNT          043076  ent     TSTS61          000000  ext
TSTDDT          032272          TSTE55          000000  ext     TSTPNX          043116          TSTS62          000000  ext
TSTDEO          034331          TSTE56          000000  ext     TSTRAO          033031          TSTS63          000000  ext
TSTDEB          034365          TSTE57          000000  ext     TSTRAA          033040          TSTS64          000000  ext
TSTDEC          034315          TSTE6           000000  ext     TSTRAN          033015          TSTS65          000000  ext
TSTDED          034354          TSTE60          000000  ext     TSTREL          047027          TSTS66          000000  ext
TSTDEE          034350          TSTE61          000000  ext     TSTREP          047020  ent     TSTS67          000000  ext
TSTDER          034347          TSTE62          000000  ext     TSTREQ          047021          TSTS7           000000  ext
TSTDIA          032265          TSTE63          000000  ext     TSTS1           000000  ext     TSTS70          000000  ext
TSTDIS          033046          TSTE64          000000  ext     TSTS10          000000  ext     TSTSAV          064653'
TSTDLO          032653          TSTE65          000000  ext     TSTS11          000000  ext     TSTSEO          032422
TSTDSO          032623          TSTE66          000000  ext     TSTS12          000000  ext     TSTSEG          032457
TSTDSA          032612          TSTE67          000000  ext     TSTS13          000000  ext     TSTSEL          032416
TSTDSL          032644          TSTE7           000000  ext     TSTS14          000000  ext     TSTSEX          032504
TSTE1           000000  ext     TSTE70          000000  ext     TSTS15          000000  ext     TSTSGO          032470
TSTE10          000000  ext     TSTE71          000000  ext     TSTS16          000000  ext     TSTSPO          032773
TSTE11          000000  ext     TSTE72          000000  ext     TSTS17          000000  ext     TSTSPC          032751
TSTE12          000000  ext     TSTE73          000000  ext     TSTS2           000000  ext     TSTSSW          032704
TSTE13          000000  ext     TSTE74          000000  ext     TSTS20          000000  ext     TSTSUB          047012  ent
TSTE14          000000  ext     TSTE75          000000  ext     TSTS21          000000  ext     TSTSWI          032672
TSTE15          000000  ext     TSTE76          000000  ext     TSTS22          000000  ext     TSTSWP          032677
TSTE16          000000  ext     TSTEBF          200000  spd     TSTS23          000000  ext     TSTTAO          032732
TSTE17          000000  ext     TSTENO          032621          TSTS24          000000  ext     TSTTAK          032715
TSTE2           000000  ext     TSTENA          032616          TSTS25          000000  ext     TSTTAN          032713
TSTE20          000000  ext     TSTEOP          033304          TSTS26          000000  ext     TSTU1           000000  ext
TSTE21          000000  ext     TSTEXE          031506          TSTS27          000000  ext     TSTU10          000000  ext
TSTE22          000000  ext     TSTEXG          031510          TSTS3           000000  ext     TSTU11          000000  ext
TSTE23          000000  ext     TSTFLG          047014  ent     TSTS30          000000  ext     TSTU12          000000  ext
TSTE24          000000  ext     TSTGET          042775          TSTS31          000000  ext     TSTU13          000000  ext
TSTE25          000000  ext     TSTHIG          047024          TSTS32          000000  ext     TSTU14          000000  ext
TSTE26          000000  ext     TSTHLO          032566          TSTS33          000000  ext     TSTU15          000000  ext
TSTE27          000000  ext     TSTHLC          032574          TSTS34          000000  ext     TSTU16          000000  ext
TSTE3           000000  ext     TSTHLL          032603          TSTS35          000000  ext     TSTU17          000000  ext
TSTE30          000000  ext     TSTHLP          032554          TSTS36          000000  ext     TSTU2           000000  ext
TSTE31          000000  ext     TSTHLT          032303          TSTS37          000000  ext     TSTU20          000000  ext
TSTE32          000000  ext     TSTINH          047023  int     TSTS4           000000  ext     TSTU21          000000  ext
```

```
DFPTA1  Test Control and Port Debugger  MACRO %53B(1242) 07:46  1-Jul-85 Page S-6
DFPTA1  MAC     23-May-85 20:41         SYMBOL TABLE                                    SEQ 0435

TSTU22           000000  ext    $GARG            000001         .MSOA            000007  spd
TSTU23           000000  ext    $ONETM           030056         .MSDO            000007  spd
TSTU24           000000  ext    $TTCHR           030231         .MSELE           000007  spd
TSTU25           000000  ext    $TWCNT           030225         .OARG            042467  ext
TSTU26           000000  ext    %ADDR            007754  spd    .PIOFF           046466  ext
TSTU27           000000  ext    %ML      775377  530000  spd    .PION            046546  ext
TSTU3            000000  ext    %MR      001000  000040  spd    .PNTOC           000000  ext
TSTU30           000000  ext    .CGOT            042630  ext    .RAND    040000  000000  spd
TSTU31           000000  ext    .CLOSE           032307  ext    .RCCER           100000  spd
TSTU32           000000  ext    .COMM            040132  ext    .RCENA   000400  000000  spd
TSTU33           000000  ext    .CONI            000000  ext    .RCJP            000060  spd
TSTU34           000000  ext    .CONO            000000  ext    .RCONT           000340  spd
TSTU35           000000  ext    .DARG            042544  ext    .RCRTN           000240  spd
TSTU36           000000  ext    .DATAI           000000  ext    .RD      001000  000000  spd
TSTU37           000000  ext    .DATAO           041121  ext    .RESET           030644  ext
TSTU4            000000  ext    .DBCMD           034776         .RJ              000000  spd
TSTU40           000000  ext    .DBDIS           035146         .RJMAP           000040  spd
TSTU41           000000  ext    .HLCMD           035335         .RLDLM           230000  spd
TSTU5            000000  ext    .INPNT           034376  ext    .RMGC            000000  spd
TSTU6            000000  ext    .INWD1           031003  ext    .ROENA           000000  spd
TSTU7            000000  ext    .ISWT            055561  ext    .ROR     030000  000000  spd
TTALTM   037340  000003         .JBSA            000120  spd    .RRDLM           220000  spd
TTIDEC   037200  000003         .LADDR   000100  000000  spd    .RSOA    400000  000000  spd
TTIME            045433         .LAND            000000  spd    .RSDO    700000  000000  spd
TTNBRF           030510         .LCCER           000000  spd    .RSELE           005000  spd
TTPN0            045456         .LCENA           000000  spd    .RSWIT           030634  ext
TTPN1            045463         .LCJP            000000  spd    .SARG            042063  ext
TTPN2            045470         .LCONT           000000  spd    .STCMD           035415
TTPNT            045440  ent    .LCRTN           000000  spd    .STDIS           035431
TTPNX            045507         .LD              000000  spd    .SWCHP           055566  ext
TTSIXB   037400  000003         .LJ              010000  spd
TTYYES           032474  ext    .LJMAP           000000  spd
TUSER    001000  000000  spd    .LLDLM           000000  spd
TWORD            052000  int    .LMGC            000001  spd
TXALL    060000  000000  spd    .LOENA           002000  spd
TXTINH           000200  spd    .LOR             000000  spd
UDEBUG           064654' int    .LRDLM           000000  spd
UNAME            035643  ext    .LSOA            000000  spd
USER             030037         .LSDO            000000  spd
USRION           032322  ext    .LSELE           000000  spd
UUT              047006  ent    .LSWIT           000000  ext
WRDERR           042621  ext    .MAND            000007  spd
XWORDL           064655'        .MCCER           000037  spd
XWORDR           064656'        .MCENA           000001  spd
XXW              064657'        .MCJP            000017  spd
XXX              064660'        .MCONT           000017  spd
ZAL      200000  000000  spd    .MCRTN           000017  spd
ZCB      400000  000000  spd    .MD              000007  spd
ZEB              000000  spd    .MJ              007777  spd
ZMP      300000  000000  spd    .MJMAP           000017  spd
ZSE      100000  000000  spd    .MLDLM           000037  spd
$$TOGG           030521         .MMGC            001777  spd
$ARG2            000001         .MOENA           000001  spd
$B               000044         .MOR             000007  spd
$CHR             000044         .MRDLM           000037  spd
```

```
                                                                                                      SEQ 0436
AADDR       111#
AAPNT        45     8229#
AAPNT0     8255     8261#
AAPNT1     8262     8273#
AAPNTX     8257     8264     8269     8278#
ABORT       259     1786
ADRPE      9341
AEBUF        50     8419     8449     8459     8511#    8551     8582
AEXEC        45     8123#
AEXEC0     8165     8178#
AEXECX     8144     8160     8195#
ALACT        46     8133     8189     8215#    8249
ALCOR        46     8129     8190     8214#    8239     8245     8266
ALCSR        46     8132     8161     8211#    8236
ALEBF        46     8181     8212#
ALERR        46     8128     8213#
ALFLC        47     8136     8163     8200     8220#    8273
ALFLE        47     8135     8191     8196     8218#    8261
ALFLS        47     8134     8143     8159     8198     8216#    8254
ALFLT        47     8137     8166     8202     8222#
ALL         718
ALSRT        46     8130     8210#    8231
ALTF         38      255      322#     322      335      339     1040     1868     1898     2861     2865     4447     4504     4593
           4909     4937     5027     5081     5317     5371     5836     6311     6355     6412     6440     7286     7369
ALTIM        46     8127     8153     8209#
ALTMGO      338     1602     1648     1686     1869     1872     2864     4507     4597
ALU         721
ALUFR      5000     5040     5114     5179     5218     5281#
ALUTO      5002     5042     5116     5181     5220     5282#
ANEXT      4498#    4567     4694     4887     4913     4940     5078     5145     5163     5241     5375     5427     5428     5449
           5533     5859     5866     5873     5908     5915     5923
ARGFLG      123#     414      417      667      675      932     1448     1612     1620     1691     1699     1731     2895     2898
           4519     5161     5447     5574     5617     5992     6039
ARGUM       123#     673      941     1001     1016     1201     1242     1363     1365     1459     1558     1564     1618     1697
           3261     3263     3431     3473     3499     3565     3590     3605     3633     3639     3684     3728     3737     3830
           3870     3900     3996     4078     4079     4084     4152     4157     4296     4310     4349     4429     4439     4526
           4534     4542     4550     4558     5011     5019     5108     5123     5164     5190     5198     5301     5309     5399
           5413     5450     5476     5484     5585     5628     5673     5683     5746     5756     5817     5818     5823     5828
           5837     5918     5926     6003     6050     6091     6101     6159     6169     6239     6274     6459     6568     6595
           6652     6657
BBPN0      8537     8547#
BBPN1      8522     8557#
BBPN2      8568     8578#
BBPNT        49     8519#
BBPNX      8553     8587#
BEGIN     10213
BEXCAL     8360     8482#
BEXCHK     8361     8496#
BEXEC        49     8345#
BEXERR     8363     8364     8365     8366     8370#
BEXEX1     8352     8357     8371#
BEXJRS     8362     8469#
BEXSS0     8407#    8430
```

```
                                                                                                                  SEQ 0437
BEXSSC    8359    8380#
BEXSSS    8358    8378#
BFLAG     8455    8458     8513#    8533     8564
BONETM     167#    167      169      387
BUFCOM     124#   6717     9251     9365
BUFF        22    6626     6659     6661     7065     9035     9036     9037     9046    10123#
BUFGEN     124#   6623     9043
CADDR       27    4490#    4515     4579     4583     4881     4898     4900     4928     4932     4952     4964     4978     4985
          5597    5640     5709     5784     6015     6062     6127     6197     6322     6376     6406     6434     6572     6596
          6599    7137     7176     7601     7621
CALL      7494
CALMAR    7417#   7417     7419     7443
CALPAR      26    4960     6023     6204     6410     6438     7148     7280     7416#
CBASE      114#    195
CBUF        67    6624     9044    10134#
CBUFSZ    6624    9044    10134
CBUS       723
CCMIC     6483    6484     6551#
CCPN0     9299    9312#
CCPN2     9314    9330#
CCPN3     9332    9337#
CCPN4     9335    9360#
CCPNT       55    9285#
CCPNX     9361    9369#
CCSR        56    9183     9229     9267#    9307
CCTAB1    6509    6518     6530#
CCTAB3    6496    6498     6508     6517     6547#
CCWPNT     114#   6685
CDERR       56    9253     9272#    9360
CEBUAD    9178#   9178     9192
CEBUF       50    8386     8460     8512#    8548     8579
CEBUFA      56    9190     9196     9232     9270#    9313     9325
CEBUFC      56    9191     9195     9197     9269#    9312     9319
CEXCAC    9003    9151#
CEXCAL    9002    9137#
CEXCOM    9005    9247#
CEXEC       55    8986#
CEXERR    8997    9008     9009     9013#
CEXEX1    8992    9014#
CEXEXI    9007    9166#
CEXJRS    9006    9124#
CEXSRD    8998    9021#
CEXSRW    9000    9084#
CEXSTR    9001    9097#
CEXSWR    8999    9072#
CEXWA2    9193    9196#
CEXWA3    9182    9223#
CEXWA4    9212    9227#
CEXWAI    9004    9176#
CHDATA     114#   9213
CHINIT     114#   6625     9045
CHKARG     123#    609     1048     1059     1074     1488     1518     1524     1535     2814     3099     3116     3193     3211
          3216    3767     3811     3840     3852     3882     3940     3951     3961     4027     4176     4248     4271     4294
          4320       4359       5906       5934       6298       6334       6396       6425       6469       6613       6640       6683       6694       6708
          6728       6742       6798
CHKCSR     110#
CHKTS0    6964       6974#
CHKTS1    6978       6987#
CHKTST    6936       6961#
CHKTSX    6970       6982       6986       6988#
CIPORT     238        241        243       1107       1108       1137       1150
CISEL       67        229        239       1154       1180       1216       1218       1228       1233       1263       1268       1286       1293       1307
          1311       1332       1340       1765       1775      10084#
CLEN        56       9023       9040       9047       9063       9074       9086       9249       9274#      9363
CLOGBF    9217       9218       9279#      9333
CLRBUF     120#       346       2859       4601       4654       4674
CLREBU     110#      7123
CMD0      2820#      2910
CMD1      2829#      2863       2889       2907       2909       2916
CMD2      2837#
CMD3      2860#      2875       2881       2893
CMD3A     2878#      2892
CMD4      2876       2890#
CMDQUE    2884       2914#
CMDQUU    2915       2918#
CONFIG     224        605       1098#      3110
CONPNT     611       1158       1166#      3118
CONSW    10194
CONVSX     121#      2587       4664
CPAT        56       9026       9038       9060       9077       9089       9092       9250       9273#      9364
CRAMFR    4421       4430       4444       4455       4466       4488#      4514       4566       4578       4582       4598       4693       4880       4886
          4897       4912       4927       4939       5861       5910
CRAMPE    8162
CRAMTO    4419       4431       4440       4450       4468       4489#      4568       4695       4888       4914       4941       5860       5909
CSERR     9210       9216       9235       9271#      9337
CSETCC    9029#      9080       9093
CSHFLG     156
CSHMEM     157
CSREN0    7528#      7542
CSREN1    7533       7540#
CSREN2    7530       7541#
CSRENB    7540       7550#
CSRENG      31       7510       7522#
CSRENS    7523#      7523       7566       7586
CSRENX    7528       7543#
CSRPNT      31       3815       4013       4043       4261       6460       7501#
CSRPNX    7507       7511#
CSTATF      56       9211       9277#
CSTWRD      67       9217      10136#
CTIME     9268#
CTRAN     9028       9039       9048       9055       9079       9091       9248       9275#      9362
CURTIM     120#      1657
CWORDL      27       4492#      4512       4561       4576       4725       4727       4729       4731       4733       4735       4764       4766       4768
          4770       4772       4774       4878       4904       4934       4956       4983       5598       5601       5641       5644       5710       5713
          5785       6016       6021       6024       6063       6128       6198       6202       6205       6324       6343       6348       6383       6403
          6431       6570       6576       6597       6600       7154       7169       7180       7603       7623
```

```
CWORDR      27     4493#   4513    4564    4577    4737    4739    4741    4743    4745    4747    4749    4751    4753       SEQ 0439
          4755     4757    4759    4776    4778    4780    4782    4784    4786    4788    4790    4792    4794    4796
          4798     4879    4905    4935    4957    4990    5599    5602    5604    5642    5645    5647    5711    5714
          5717     5786    5788    6017    6022    6025    6064    6129    6199    6203    6345    6351    6384    6404
          6432     6571    6577    6598    6601    7168    7170    7182
D1LIT    10212#
DACRAM    3150     4395#
DBCRAM    3149     4387#
DBPNT     3205     6649#
DBPNT0    6660#    6673
DBPNT1    6665     6670#
DBREA0    5999#    6031
DBREAK    3174     5992#
DC1ST     4477#    4477    4508    4510    4588    4591
DCBR0     6090     6100    6106#
DCBR1     6127#    6137
DCBRK     3176     6082#
DCCOD1    6480#    6499
DCCOD2    6482     6503#
DCCOD3    6505#    6510
DCCOD4    6507     6512#
DCCOD5    6514#    6519
DCCOD6    6516     6520#
DCCODE    3187     6468#
DCLEAR    3129     3950#
DCMAR0    5672     5682    5688#
DCMAR1    5709#    5724
DCMAR2    5716     5722#
DCMARK    3171     5664#
DCODPB    4685     4764#
DCOLDB    4633     4725#
DCOM0     4428     4438    4444#   5862    5911
DCOM1     4477#
DCOM3A    4624     4629    4658#
DCOMA0    4607#    4644
DCOMA1    4610#    4680    4688
DCOMA2    4632     4637#   4657    4677
DCOMA3    4625     4638#   4641    4660
DCOMA4    4666     4684#
DCOMA5    4661     4686#
DCOMAA    4620     4626#   4652
DCOMAG    4580     4588#
DCOMAL    4482     4582#   4700
DCOMAX    4612     4648    4692#
DCOMDB    4481     4576#   4699
DCOMDE    4480     4504#   4569
DCOMEX    4479     4897#   4915
DCOMEZ    4911     4916#
DCOMGO    4373     4381    4389    4397    4405    4413    4418#
DCOML0    4927#    4942
DCOMLI    4484     4925#
DCOMLX    4938     4943#
DCOMPA    3208     6707#
```

```
DCOMSG    4637    4704#
DCOMZE    4483    4878#   4889
DCONI     3163    3851#
DCONI1    3854#   3857
DCONO     3164    3864#
DCONO1    3870#   3873
DCONT     3133    4247#
DCWPNT    3206    6682#
DDALU     3156    5102#
DDALU0    5108#   5165    5919
DDALUM    5136    5140    5141    5267#
DDALUX    5162    5169#
DDATI     3165    3881#
DDATI1    3884#   3886
DDATO     3166    3894#
DDATO1    3900#   3903
DDCRAM    3148    4379#
DDCSR     3125    3824#
DDEBU1    4297    4305#
DDEBUF    3141    4299#
DDELA0    7061#   7072
DDELAX    7064    7073#
DDELAY    7056#   9180
DDLOC0    5399#   5451    5927
DDLOCM    5426    5429    5551#
DDLOCS    3160    5393#
DDLOCX    5448    5455#
DDNEX0    5890    5891    5892    5893    5894    5895    5905#
DDNEX1    5896    5897    5898    5915#
DDNEX2    5899    5900    5901    5923#
DDNEXT    3109    5882#
DDRAR     3145    4340#
DDTAB     5883    5885    5890#
DEALU     3155    5000#
DEALU0    5010    5018    5024#   5869
DEALU1    5053#   5086
DEALUM    5056    5065    5066    5262#
DEALUX    5083    5090#
DEBTIM     120#   7025    7029    7059    7063
DEBUG1     210     358    2843
DEBUG2     212     360    2845
DEBUG3     362    2847
DEBUG4     214     364    2849
DEBUG5     216     366    2851
DECRAM    3151    4371#
DECSR     3124    3810#
DECVER      10    7686    7810
DECYN      123#   3518    3542
DEEBUF    3140    4270#
DELAR     3144    4319#
DELOC0    5300    5308    5314#   5876
DELOC1    5344#   5377
DELOCM    5355    5356    5545#
```

```
DELOCS   3159    5290#
DELOCX   5373    5381#
DENE0    5827#   5840
DENEX0   5844    5845    5846    5847    5848    5849    5859#
DENEX1   5850    5851    5852    5866#
DENEX2   5853    5854    5855    5873#
DENEXT   3108    5817#
DESTA0   5948#   5968
DESTAK   3188    5933#
DESTAX   5950    5972#
DESTAY   5957    5960    5973#
DESTKM   5938    5979#
DETAB    5832    5834    5844#
DEVREL    115#    267    1050    1076
DEVREQ    115#    460    1130    1138    1799    2823
DFEXA0   6429#   6443
DFEXAM   3184    6424#
DFEXAX   6430    6441    6445#
DFILLN   3189    6561#
DFILLP   3190    6588#
DFILN0   6572#   6579
DFILP0   6599#   6604
DFLAG    4372#   4372    4380    4388    4396    4404    4412    4478    4630    4697    5044    5126    5222    5334
         5416    5508    5831    5833    5882    5884
DFLIS0   6401#   6415
DFLIST   3183    6395#
DFLISX   6402    6413    6417#
DFLOA0   6304#   6314
DFLOAD   3181    6297#
DFLOAX   6305    6312    6316#
DFVER0   6341#   6358
DFVERF   3182    6333#
DFVERR   6353    6371#
DFVERX   6342    6356    6360#
DFVERY   6374    6387#
DHELP    3097    3253#
DHELP0   3271#   3275
DHELP1   3271    3298#
DIINIT   3201    6797#
DIPRIN   3191    6741#
DISGO     473     592#    622     634
DISLIS    422     442     523#
DISP0     330#    333     397     402     412     449     459     463     469     480     487
DISP0A    399#    411
DISP0B    398     409#
DISP1     421#
DISP2     381     425     473#
DISPAT    254     322#
DISPD     431     453#
DISPE     408     456     467#
DISPR     432     439#
DISPX     435     478#
DISQUE    407     485#
```

```
DISQUU     486      489#
DLBR0     6158     6168     6174#
DLBR1     6197#    6225
DLBR2     6206     6220#
DLBRK     3177     6150#
DLBRX     6222     6226#
DLCRAM    3152     4411#
DLGPNT    3207     6693#
DLIST     3180     6268#
DLMAR0    5745     5755     5761#
DLMAR1    5784#    5809
DLMAR2    5790     5804#
DLMARK    3172     5737#
DLMARX    5806     5810#
DLOAD     3179     6233#
DMARK     3169     5574#
DMARK0    5581#    5609
DRBR0     6046#    6070
DRBRK     3175     6039#
DRCRAM      26     4584     4902     4929     4977#    5600     5643     5712     5787     6018     6065     6130     6200     6323
          6347     7602
DRDINC    6617#    6643
DRDINI    3203     6612#
DRESET    3128     3939#
DRMAR0    5624#    5652
DRMARK    3170     5[illegible]7#
DROUT1    3914#    3931
DROUTN    3167     3910#
DSADDR    3474     3480     3485     3795#    3963     3991
DSDATA    3500     3505     3796#    3965     3999
DSET      3105     3422#
DSETA1    3472     3483#    3664     3967
DSETA2    3484#
DSETAD    3453     3469#
DSETB1    3683     3693#
DSETC1    3498     3503#
DSETC2    3504#    3665     3968
DSETCS    3454     3495#
DSETD1    3589     3612#
DSETD2    3613#    3669
DSETD3    3617     3621#
DSETDA    3458     3586#
DSETDX    3598     3607#
DSETE1    3541     3549#
DSETE2    3550#    3667
DSETEB    3456     3538#
DSETF0    3638     3641#
DSETF1    3632     3645#
DSETF2    3646#    3670
DSETFA    3463     3677#
DSETFN    3459     3629#
DSETHE    3462     3766#
DSETHL    3770     3773#
```

```
DSETL1   3564     3574#
DSETL2   3575#    3668
DSETLN   3457     3561#
DSETP1   3517     3525#
DSETP2   3526#    3666
DSETPL   3460     3721#
DSETPP   3727     3736     3745#    3758
DSETPR   3455     3514#
DSETPX   3756     3759#
DSETUP    115#    1099
DSETWH   3461     3659#
DSINIT   3138     3960#
DSPAT    3596     3614     3724     3803#    6618
DSPEAR   7653     7715
DSSCLR   3137     3969     4175#
DSSPNO   4160#    4167
DSSPNT   3136     4151#
DSST0    4091#    4131
DSST1    4121#
DSSTEP   3134     4075#
DSSTX    4105     4110     4130     4135#
DSTAR0   3995#
DSTART   3131     3988#
DSTOP    3132     4026#
DSTOP1   4036     4041#
DSTPAT   3606     3619     3804#    6619
DSWIT    3223     3224     3225     3226     3227     3228     3229     3230     3231     3232     3233     3234     3235     3236
         3237     3238     3239     3240     3241     3242     3243     3244     3245     3246     3247     6727#
DSWORD   3571     3576     3801#    6621     6649     6716
DTRAC0   4073     4078#
DTRACE   3135     4071#
DTRANS   3186     6453#
DTRFLG   4072     4075#    4075     4086     4104     4116     4124
DWCRAM     26     4565     4692     4882     4951#    5607     5650     5720     6029     6068     6133     6308     6578     6602
         7624
DWRINI   3204     6639#
DZALU    3157     5179#
DZALU0   5189     5197     5203#
DZALU1   5231#    5252
DZALUM   5232     5236     5237     5276#
DZALUX   5249     5256#
DZCRAM   3153     4403#
DZCSR    3126     3839#
DZEBUF   3142     4293#
DZLOC0   5475     5483     5489#
DZLOC1   5517#    5535
DZLOCM   5518     5519     5560#
DZLOCS   3161     5465#
DZLOCX   5531     5539#
DZRAR    3146     4358#
EADDR     111#
EBUS      719
ENATST   1441#    1441     1445#    1445     1476     1478
```

```
ENDFIX      11
ERESET     110#     3942
ERFLG      345       730      1831      2534      2539      2547      2831      6839      7485      7708      7724      7792      8773      8786
          8793      8795      8809      8829      9169      9198      9215      9223      9238      9255      9256      9757      9797      9802
          9804      9831      9837      9842      9852     10165     10173
ERRCNT    1804#     1804      1836      1844
ERRPC     6840
EXE      10193#
EXEN     10190#
FGETW      121#     6304      6341      6401      6429
FINCMD     121#      396      2874      4622      4627
FINECH      32       162#      162       389       390      1551      1553      2867      2868
FINIT      121#     6301      6338      6398      6427
FINPUT      32       161#      161       388       395      1587      1733      2866      2873      4621      4626
FIOFF      121#      336       448       468       944       950      1007      1022      1207      1248      1369      1450      1462      1468
          1569      1897      2862      2888      2906      3267      3429      3435      3478      3569      3594      3603      3688      3742
          3749      3828      3868      3898      4003      4006      4031      4034      4097      4100      4252      4255      4276      4303
          4308      4344      4448      4453      4457      4505      4524      4532      4540      4548      4556      4594      4656      4676
          5028      5033      5037      5106      5111      5121      5206      5211      5215      5318      5323      5327      5397      5403
          5411      5492      5497      5501      5576      5588      5619      5631      5691      5696      5700      5764      5769      5773
          5956      5959      5994      6006      6041      6053      6109      6114      6118      6177      6182      6186      6237      6242
          6248      6253      6272      6277      6283      6288      6457      6565      6592      6714
FLDLIS    4671      4803#
FLTCH0    2532#     2538
FLTCH1    2531      2545#     2551
FLTCHK    1830      2526#
FLTCHX    2539      2544      2555#
FLTCLS   10111#
FLTHA0    2496#     2518
FLTHAN    1857      1862      2494#
FLTNUM    2533      2546      3690      3694      3696     10112#
FLTPNT     749#      749       751      1805
FMTTYP     166#      166
FOARG      121#
FORPNT     121#      183      1919
GENCCW     114#     6629      9052
GETLOG     114#     6697
GETPPN    7658      7779      7823#
HAVECI     116#    10200
HAVENI     116#    10201
IADDR       59      9660      9688      9690      9703      9873#     9901      9910
IADDRF    9778      9848      9874#     9904      9913
ICSR        59      9769      9845      9875#     9916
IDEBUG      19       143#      143       208       217       356       367       815       822      1829      1847      1860      1863      2500
          2841      2852
IEBUF     9787      9792      9888#     9919
IEXCAC    9503      9739#
IEXCAL    9502      9725#
IEXEC       58      9485#
IEXERR    9496      9508      9509      9513#
IEXEX1    9491      9514#
IEXEXI    9507      9754#
IEXJRS    9505      9712#
```

```
IEXSDE    9498      9555#
IEXSEI    9501      9623#
IEXSEX    9497      9521#
IEXSID    9499      9589#
IEXSIN    9500      9658#
IEXSTR    9506      9683#
IEXWA0    9784      9799      9812#
IEXWA1    9815      9827#
IEXWA2    9829      9836#
IEXWA5    9768      9842#
IEXWA6    9832      9838      9843#
IEXWAI    9504      9764#
IEXWRG    9786      9791      9861#
IFLAG       59      9537      9540      9571      9574      9605      9608      9640      9643      9665      9674      9782      9827      9877#
          9963      9969      9979
IIADR     9523      9532      9543      9557      9566      9577      9591      9600      9611      9625      9635      9646      9798      9800
          9881#    10002     10024     10046
IIDEPA    9528      9565      9596      9630      9671      9887#    10043     10058
IIDEPC    9527      9563      9580      9595      9629      9670      9803      9808      9886#    10042     10052
IIEXAA    9531      9562      9594      9634      9669      9788      9793      9795      9885#    10021     10036
IIEXAC    9529      9546      9561      9593      9628      9649      9668      9796      9884#    10020     10030
IIINCA    9526      9560      9599      9633      9667      9806      9883#     9999     10014
IIINCC    9525      9559      9597      9614      9631      9666      9801      9882#     9998     10008
IINTPC    9814      9822      9880#     9992
IIOPF       59      9878#     9925      9930      9937      9941      9952      9955      9958
IIPN0     9908      9924#
IIPN1     9963#
IIPN2     9967      9978      9984#
IIPN3     9928      9986      9997#
IIPN4    10000     10019#
IIPN5    10022     10041#
IIPNT       58      9894#
IIPNX    10044     10063#
IITYPE    9813      9820      9879#     9984
INDLIS      36       511#     1506      1850      2578      2624      6826      6997
INHFLT    1710
INITPD     112#     1063
INITPI     112#      252       374      6800      9658
INTAPR     113#     6782
INTCON     113#     6779
INTCSR     113#     6785
INTEND     113#     3198      9685
INTNUM     113#     3196      3197      3199      6745      6755      6758      6843      9684      9812      9816      9830
INTPC      113#     6769      9821
INTTIM     113#     6772
INTTYP     113#     6760      6788      9819      9836
INTUSE     113#     6775
IPACLR     110#     3914      3953      7472      7598      7616      8141      8172      8705      9097      9683
IPAER     9349
IPASRT     111#     4001      4250
IPASST     111#     4095      5954
IPASTP     111#     4029
IPRINO    6745#     6749
```

```
IPRINI     6747     6755#
IPRINT       63     6744#
IPRINX     6746     6750#
ISOPN0      757#     761      765
ISOPN1      759      766#
ISOPN2      771#     777
ISOPN3      775#     781      787      791
ISOPN4      783      788#
ISOPN5      773      795#
ISOPN6      809#     824
ISOPN7      816      821#
ISOPN8      811      828#
ISOPNT      735      746#    1769     1782     1899
ISOPNX      766      829#
ISTOP       110#    3995     4083     4273     4305     4322     4346     4361     4473     5593     5636     5705     5778     5944
           6011     6058     6123     6191     6255     6303     6340     6567     6594     7122
ITERAT      218
ITIME      9876#
ITRCNT      219
LARG2      5124     5154     5414#    5414     5440
LASARG      123#    3607     3659
LASDTC     6617     6632#    6710
LAST       7494
LAXER      9347
LDCRAM      109#    4962     4969     7151     7164
LDCSR       109#    3831     3843     3871     4324     4348     4363     5073     5153     5157     5246     5363     5365     5439
           5443     5526     5528     6491     6493     7961     8149     8168     8179     8399     8414     8426     8716     8800
           8811     9033     9112     9188     9203     9700     9775     9865     9867
LDEBUF      109#    3901     4311
LDEBUG       18      142#     142      207      215      355      365     2840     2850     7461
LDRAR       109#    4350     4364     4955     4967     4981     4988     5071     5151     5244     5361     5437     5524     6489
           7141     7161     7956     8147     8409     8709     9031     9104     9692     9863
LGWC       9351
LOCFR      5290     5330     5405     5425     5465     5504     5565#
LOCTO      5292     5332     5406     5467     5506     5566#
LOGBUF      114#    6696
LOGPNT      114#    6698     9334
LOOPGM      265
LOOPTS     1605
MAPNEW      144
MBARG       674#     674     1725
MBARGF      666#     666      676     1723
MBCN        459     1112     1120     1128     1136     1212     1253     1274     1297     1300     1323     1325     1327     1331
           1334     1343     1345     1347     1795     1797     1798     1824     2822     3112     3121     6855     6859     7463
           7467     7476     7478    10202
MBUS        701      724     1722
MCBUS      6977     6983
MCNVER       10     7686     7810
MCPUS       152#     152
MDEBUG       16      140#     140      206      213      353      361     2838     2846     7489
MLIST      6290     7260#
MLISTO     7263#    7287
MLISTX     7265     7288#
```

```
MLNUM        6259     7197#    7197
MLOADO       7129#    7189
MLOAD1       7166     7172     7175     7187#
MLOADN         26     5067     5142     5238     5357     5430     5520     5939     6485     7097#    7473
MLOADX       7131     7193#
MLOADY         26     6256     7099#
MMPROC       6977     6979
MONTYP       7037     7041     7043     7646     7697
MPER         9339
MPROC         722
MPRUN        5072     5152     5245     5362     5438     5525     6490     7940     7943     8148     8164     8171     8393     8396
             8398     8424     8711     8714     9032     9106     9109     9694     9697     9864
MSG          7493
MULFLG        369#     369     2854     4910     4916     4943     5082     5090     5372     5381     5827     5830     7370
MULINI        370#     370     2855     4925     5839
MVCOM        7102     7106     7111#
MVFLAG       7100#    7100     7105     7150     7152     7163     7165
MVNUM        6337     6364     6372     7111#    7111     7173     7193     7211     7214
MVPNT        7196     7211#    7494
MVPNT1       7221#    7242
MVPNTW       7227     7231     7237     7247#
MVPNTX       7239     7244#
MVREP        7097#    7097     7099     7195
MWBADR       7116     7117     7177     7201#    7221
MWBLAC       7181     7203#    7229     7234
MWBLCO       7178     7202#    7225     7233
MWBRAC       7118     7183     7205#    7230     7236
MWBRCO       7179     7204#    7226     7235
MWDQUE       4650     4825#
NCBUS         841#     895      896      897      898      899      900      901      902      903      904      905      906      907
              908      909      910      911      912      913      914      915      916      917      918
NDCB         6980     6984
NDMP         6980
NEBUS         839#     846      847      848      849      850      851      852      853      854      855      856      857      858
              859      860      861      862      863      864      865      866      867      868      869
NERROR      10113#
NETBAD        760      837#    1801     1802     1803     2497
NETDES        798      813      846#    2502
NETPNT        753      754      755      762      763      778      779      784      785      788      789      790      797      810
              818      836#
NEXM         9345
NIPORT        233      236      243     1107     1108     1129     1144
NISEL          67      228      234     1148     1170     1213     1215     1220     1225     1255     1260     1275     1282     1298
             1304     1310     1328     1339     1763     1773    10083#
NMBUS         842#
NMPROC        840#     873      874      875      876      877      878      879      880      881      882      883      884      885
              886      887      888      889      890      891
NOTWCO       9343
NSSTEP         43     7932     7950     7975     8042#    8062     8079     8383     8403     8437     8530     8561
ODELA0       7027#    7030
ODELA1       7018     7036#
ODELAY         31     7017#    8154     8784     9766
OPRSEL        226      325      331      334      380      479     1716     1901     1913
```

```
OVN        9355
P          1634       1637       1638       1639       2615       2616       2816       3113       3122       6867       6938       6939       6987       7108      SEQ 0448
           7194       7475       7481       7490       7992       7993       7994       8008       8022       8036       8204       8461       8462       8463
           8477       8491       8505       8723       8736       8750       8764       8774       8830       8831       8832       9066       9118       9132
           9146       9160       9170       9239       9240       9241       9257       9258       9259       9549       9583       9617       9652       9677
           9706       9720       9734       9748       9758       9853       9854       9855
PARFLG       27       3520       3522       3527       3529       3798#      4959       5605       5606       5608       5648       5649       5651       5718
           5719       5721       6027       6028       6030       6066       6067       6069       6131       6132       6134       6306       6307       6309
           6409       6437       7617       7618       7627
PASCNT      197        382       1914
PAT        3724       3725       3726       3734       3735       3745       3746       3747       3752       3753       3757       6618       9038       9092
           9250       9364
PAT1       3725#      3725       3729#      3729       3734       3740       3745
PAT2       3726#      3726       3735       3738       3739       3757
PATCH     10144#
PATCHO    10143#
PATPNT      124#      3615       3752
PCMD       2830#      2830       2894       2902
PFSET       115#       201
PGMNAM       20        130#
PMODE        38        220        368       1747       2853      10115#
PNTCRM     4936       6411       7281       7299#
PNTDEV      120#       796       1910       6823
PNTENB      182        344       2833
PNTHST     4117       4160       4198#
PNTWD      4906       6381       6385       6439       7382#
PORSEL       67       1756       1761       1770       1796      10085#
PORTCI       67       1120       1136       1300       1325       1331       1345       1797       6859       7467       7478      10082#
PORTNI       67       1112       1128       1297       1323       1327       1343       1795       6855       7463       7476      10081#
PRSFLG     3544       3546       3551       3553       3799#      4092
PTEST       457       1322#      2820
PTESTO     1324       1326       1330       1335#
PTESTP      458       1338#      2821
PTESTX     1342       1347#
PVPAGI      145
QADDR        56       9100       9102       9265#      9292       9301
QADDRF       56       9206       9266#      9295       9304
RALUB      5001#      5001       5012       5024       5035       5039       5054       5084       5113       5135       5180       5191       5203       5213
           5217       5231       5250       5867       5916
RALUE      5003#      5003       5013       5020       5030       5041       5085       5115       5182       5192       5199       5208       5219       5251
           5868       5917
RANCNT     1654#      1654       1667       1673
RANDBS     1630
RANGEN     1628#      1663
RANNUM     1629#      1629       1632       1658
RDCRAM      109#      4982       4989       7153       7167
RDCSR       109#      1113       1121       3813       3854       4011       7069       8158       8169       8173       8785       9181       9767
RDEBUF      109#      3884       4281       5074       5366       8180       8418       8801       9189       9868
RDLAR       109#      4325       6494       7965       8187       8427       8812       9204       9776
REL         116#     10192
RELIAB     1813
REQ1        116#     10191
REQ2        116#
```

```
RESCRO    7621#   7626
RESCRM    5093    5170    5257    5384    5456    5540    5974    6525    7615#
RESQAV    7071    9109
RFROM     4420    4467    4496#
RLOCB     5291#   5291    5302    5314    5325    5329    5340    5344    5350    5354    5374    5466#   5466    5477
          5489    5499    5503    5517    5532    5874    5924
RLOCE     5293#   5293    5303    5310    5320    5331    5376    5468#   5468    5478    5485    5494    5505    5534
          5875    5925
RSTART    1881
RTO       4418    4469    4495#
RUNTME     120#   1656    1921    7024    7028    7058    7062
S1        7667#  10151
S2        7787#  10152
SAADDR      43    7969    7984    7990    8045#   8069    8086    8431    8446    8456    8542    8573
SAVADD    1746#   1746    1751    1809
SAVCMD    2816#   2816
SAVCRO    7601#   7606
SAVCRL    7599    7619    7633#
SAVCRM    5049    5131    5227    5339    5421    5513    5937    6475    7597#
SAVFLD    4662#   4662    4663    4670
SAVHST    4102    4111    4219#
SAVMAX     439#    439     445
SAVRAR    7140#   7140    7159
SB         134#
SCOOFF     124#   6842
SCOSW      124#   8052    8233    8520    8855    9286    9330    9895    9997   10019   10041
SDATA      111#   3966    4000    4042    4093    4142    4233    4260    5951    6845    7939    7944    7960    8392
          8397    8413    8423    8710    8715    9105    9110    9111    9693    9698    9699
SDEBUG      17     141#    141     354     363    2839    2848    7920    7973    8001    8014    8028    8371    8435
          8470    8483    8497    8689    8717    8729    8742    8756    8770    8818    9014    9053    9113    9125
          9138    9152    9166    9227    9514    9538    9572    9606    9641    9672    9701    9713    9726    9740
          9754    9843
SEADDR      43    7934    7981    7991    8044#   8066    8083    8385    8443    8457    8535    8539    8566    8570
SELLAR    4323    6492    7943    8425    8714    8810    9109    9202    9697    9774
SELTST    1494    1821    6927#
SELTSX    6931    6943#
SEN11L    7723    7770    7780    7785    7813#
SENT10    7659    7661    7665    7677#  10154
SENT11    7720    7721    7740    7758    7775    7784    7800#  10161   10169
SEQ        720
SETEBU     110#   4274    4306
SETLAR     110#   7945    8142    8185
SETVEC     112#    253     375    1064    6801    9659
SEXCAL    7909    8013#
SEXCHK    7910    8027#
SEXEC       42    7895#
SEXERR    7912    7913    7914    7915    7919#
SEXEX1    7901    7906    7920#
SEXJRS    7911    8000#
SEXSSO    7954#   7968
SEXSSC    7908    7929#
SEXSSS    7907    7927#
SHWC      9353
```

```
SINCYC        5072       5245       7943       8396       9864
SLAST          111#      3998       4136       4227
SLDATA        4094#      4094       4231
SNEXT          111#      3964       3997       4038       4139       4229       5953       5967       6844       7928       7929       7954       7967       8131
              8145       8379       8380       8407       8429       8699       8704       8706       8719       9101       9689
SPCHK1        7658#     10158
SPCHK2        7720#     10166      10176
SPEAR1         185#       185       7644       7671
SPEAR2          37        186#       186       7703       7794
SPRE1X        7654       7672#
SPRE2A        7704       7713#
SPRE2B        7759#      7762
SPRE2C        7724       7774#
SPRE2X        7716       7795#
SPREP1         248       7641#
SPREP2          37        731       7694#
SRTDDT        1062      10171      10205
SRTLIS         622#       977
SSADDR          43       7930       7978       8043#      8059       8076       8381       8440       8527       8558
SSADRL        4187#      4199       4228
SSADRN        4188#      4202       4230
SSCSRL        4189#      4232
SSCSRN        4180       4190#      4205       4234
SSLOC         4115       4158       4178       4179       4185#      4220       4222       4224
SSNUM         4166       4186#      4208       4221       4226
SSPN1         8054       8075#
SSPNT           42       8051#
SSPNX         8071       8091#
START           21        135#      1081       1882      10196
STARTA          21        232        252#       263        266
STARTB         231        256        261        267#
STCLOK         120#       196
STEST        10150#
STEST0       10156      10160#
STEST1       10163      10168#
STESTP       10155      10162      10170      10178#     10183
STRT1          171        190#
SWCHPT         122#
SWCOM          122#      1539       6733
SWITT          122#       209        225        257        264        324        330        357        478       1105       1604       1703       1784       1811
              1880       1900       1912       2842       6816       6976       7005       7505       7652       7714
SWPTAB         177
SWRGT          122#       175
SWSTP         3221#      6732
SWSTT          634#      1538
TADDR           53       8698       8703       8707       8838#      8857
TADDRF          53       8814       8826       8839#      8867       8876
TAKFIL          38       1559       1565       1570       1573       1576       1581       1584       1592#
TALU           981        987       2063#      2592       2593       2606       7747
TCBUS          983        989       2136#      2594       2595       2608       7749
TCOMP         8781       8791       8794       8844#      8862
TCSRF           53       8787       8820       8840#      8884
TDENA         1477       1479       1499       6831       6850
```

```
TDISP1     960     1491     1662     1664     1712     1939#
TDISP2    1714     2192#
TEBUFA      53     8802     8808     8823     8843#    8890     8902
TEBUFC      53     8803     8807     8842#    8889     8896
TEBUS      979     1938#    2591     2604     7745
TEND       962      990     1662     2184#    2595
TEXCAC    8679     8755#
TEXCAL    8678     8741#
TEXCHO    8805     8810#
TEXCHK    8680     8780#
TEXCOO    8700     8706#
TEXCON    8677     8705#
TEXEC       52     8664#
TEXERR    8675     8684     8688#
TEXEX1    8670     8689#
TEXEXI    8682     8770#
TEXJRS    8681     8728#
TEXSST    8683     8696#
TEXSTR    8676     8701#
TFAIL     8782     8789     8792     8846#    8864     8873
TLERR     7486     7493#
TLOAD       26     7461#
TLOADE    7474     7484#
TLOADX    7462     7465     7469     7481#
TMODE    10131#
TMP1     1443#     1443     1447#    1447     1455     3634     3641
TMPROC     982      988     2100#    2593     2594     2607     7748
TPAT        32     6620    10132#
TRACE       31     6814#
TRACEO    6818     6839#
TRACET    6817
TRACEX    6851     6872#
TSEQ       980      986     2004#    2591     2592     2605     7746
TSLOD1      38      347#     347     6858     7464     7477    10203
TSLOD2      38      348#     348     6862     7468     7479    10204
TSTA1      88#     2063     2311
TSTA10     88#     2070     2318
TSTA11     89#     2071     2319
TSTA12     89#     2072     2320
TSTA13     89#     2073     2321
TSTA14     89#     2074     2322
TSTA15     89#     2075     2323
TSTA16     89#     2076     2324
TSTA17     89#     2077     2325
TSTA2      88#     2064     2312
TSTA20     89#     2078     2326
TSTA21     90#     2079     2327
TSTA22     90#     2080     2328
TSTA23     90#     2081     2329
TSTA24     90#     2082     2330
TSTA25     90#     2083     2331
TSTA26     90#     2084     2332
TSTA27     90#     2085     2333
```

```
TSTA3       88#    2065     2313
TSTA30      90#    2086     2334
TSTA31      91#    2087     2335
TSTA32      91#    2088     2336
TSTA33      91#    2089     2337
TSTA34      91#    2090     2338
TSTA35      91#    2091     2339
TSTA36      91#    2092     2340
TSTA37      91#    2093     2341
TSTA4       88#    2066     2314
TSTA40      91#    2094     2342
TSTA41      92#    2095     2343
TSTA42      92#    2096     2344
TSTA5       88#    2067     2315
TSTA6       88#    2068     2316
TSTA7       88#    2069     2317
TSTADD      36      433      952     1033     1492     1665     1715     1745     1752     1810     1840     2611     6895     6928
          7752    10099#
TSTALO    1891     1896#
TSTALT    1601     1685     1889#
TSTC1      100#    2136     2409
TSTC10     100#    2143     2439
TSTC11     101#    2144     2411
TSTC12     101#    2145     2443
TSTC13     101#    2146     2444
TSTC14     101#    2147     2445
TSTC15     101#    2148     2446
TSTC16     101#    2149     2447
TSTC17     101#    2150     2448
TSTC2      100#    2137     2410
TSTC20     101#    2151     2449
TSTC21     102#    2152     2450
TSTC22     102#    2153     2451
TSTC23     102#    2154     2452
TSTC24     102#    2155     2453
TSTC25     102#    2156     2454
TSTC26     102#    2157     2455
TSTC27     102#    2158     2456
TSTC3      100#    2138     2434
TSTC30     102#    2159     2457
TSTC31     103#    2160     2458
TSTC32     103#    2161     2459
TSTC33     103#    2162     2460
TSTC34     103#    2163     2461
TSTC35     103#    2164     2462
TSTC36     103#    2165     2463
TSTC37     103#    2166     2464
TSTC4      100#    2139     2435
TSTC40     103#    2167     2468
TSTC41     104#    2168     2469
TSTC42     104#    2169     2470
TSTC43     104#    2170     2471
TSTC44     104#    2171     2472
```

```
TSTC45     104#   2172   2473
TSTC46     104#   2173   2474
TSTC47     104#   2174   2475
TSTC5      100#   2140   2436
TSTC50     104#   2175   2476
TSTC51     105#   2176   2477
TSTC52     105#   2177   2478
TSTC53     105#   2178   2479
TSTC54     105#   2179   2480
TSTC55     105#   2180   2481
TSTC56     105#   2181   2482
TSTC57     105#   2182   2483
TSTC6      100#   2141   2437
TSTC60     105#   2183   2484
TSTC7      100#   2142   2438
TSTCLS      36     441    978    1505   1849   2581   2590   2603   2627   6825   6910   6996   7728   7744
         10094#
TSTCNT    1820    1876  10110#
TSTD0     1692    1703#
TSTD1     1724    1730#  1738
TSTD10    1837    1844#
TSTD11    1828    1831   1868#
TSTD12    1823    1826   1878#
TSTD2     1726    1732   1735   1741#
TSTD3     1621    1666   1717   1745#
TSTD4     1751#   1790
TSTD5     1760#   1777   1800   1822   1884
TSTD6     1772    1782#
TSTD7     1778    1795#
TSTD8     1819#   1885
TSTD9     1824#   1877
TSTDDT     615    1058#  3103
TSTDE0    2587#   2630
TSTDEB     604    2813#
TSTDEC     430     942   1460   2571#  6240   6275
TSTDED    2580    2622#
TSTDEE    2588    2617#  2626
TSTDER    2599    2616#
TSTDIA     614    1047#  1902
TSTDIS     726    1685#
TSTDL0    1494#   1500   1511
TSTDS0    1455#   1481
TSTDSA     619    1441#
TSTDSL     620    1487#
TSTE1       71#   1939   2193
TSTE10      71#   1946   2203
TSTE11      72#   1947   2207
TSTE12      72#   1948   2208
TSTE13      72#   1949   2304
TSTE14      72#   1950   2209
TSTE15      72#   1951   2210
TSTE16      72#   1952   2211
TSTE17      72#   1953   2215
```

```
TSTE2           71#     1940    2194
TSTE20          72#     1954    2216
TSTE21          73#     1955    2217
TSTE22          73#     1956    2353
TSTE23          73#     1957    2221
TSTE24          73#     1958    2222
TSTE25          73#     1959    2354
TSTE26          73#     1960    2355
TSTE27          73#     1961    2356
TSTE3           71#     1941    2195
TSTE30          73#     1962    2357
TSTE31          74#     1963    2358
TSTE32          74#     1964    2359
TSTE33          74#     1965    2360
TSTE34          74#     1966    2361
TSTE35          74#     1967    2305
TSTE36          74#     1968    2306
TSTE37          74#     1969    2307
TSTE4           71#     1942    2196
TSTE40          74#     1970    2348
TSTE41          75#     1971    2349
TSTE42          75#     1972    2362
TSTE43          75#     1973    2363
TSTE44          75#     1974    2364
TSTE45          75#     1975    2365
TSTE46          75#     1976    2366
TSTE47          75#     1977    2367
TSTE5           71#     1943    2197
TSTE50          75#     1978    2368
TSTE51          76#     1979    2369
TSTE52          76#     1980
TSTE53          76#     1981    2234
TSTE54          76#     1982    2192
TSTE55          76#     1983    2374
TSTE56          76#     1984    2375
TSTE57          76#     1985    2235
TSTE6           71#     1944    2201
TSTE60          76#     1986    2397
TSTE61          77#     1987    2398
TSTE62          77#     1988    2399
TSTE63          77#     1989    2400
TSTE64          77#     1990    2401
TSTE65          77#     1991    2402
TSTE66          77#     1992
TSTE67          77#     1993
TSTE7           71#     1945    2202
TSTE70          77#     1994    2403
TSTE71          78#     1995    2404
TSTE72          78#     1996    2405
TSTE73          78#     1997    2370
TSTE74          78#     1998    2371
TSTE75          78#     1999    2372
TSTE76          78#     2000    2373
```

```
TSTEBF     5072      5364      8178      8396      8398      8424      8799      9187      9866
TSTENO     1444      1448#
TSTENA      618      1445#
TSTEOP     1767      1907#
TSTEXE      593       594       595       596       597       598       718#
TSTEXG      703       726#
TSTFLG       36      1475      1498      6830      6849      6898      6907      6909     10098#
TSTGET     1034      1473      6250      6285      6893#     6935      7753
TSTHIG     1845      2530      6904     10106#
TSTHLO     1373#     1377
TSTHLC     1366      1380#
TSTHLL     1373      1388#
TSTHLP      617      1356#
TSTHLT      616      1073#
TSTINH       38       747       931      1490      1611      1653      1709      1711      1713      1827     10105#
TSTL1       953#      953       961       984       993      1009      1013      1029      1036
TSTL2       930       954#      954       963       991      1014      1025      1031
TSTL3       985#      985      1024
TSTLI0      937#      956
TSTLI1      933       960#
TSTLIS      624       929#
TSTLOW     1858      2543      6906     10107#
TSTLSS      955       964      1000      1015      1029#     1038
TSTLST      625       626       627       628       629       977#
TSTMBO      668       680#
TSTMBU      599       666#
TSTMIC       37      6251      6286      6856      6860      6900     10100#
TSTMSK       37       702       725      1721      6968     10101#
TSTMUL      746      1610      1652      1690      1883      1907      6963     10104#
TSTNAM       36      6902      7009      7754     10093#
TSTNUM       36      1508      1852      2589      6828      6908      6999      7734      7737      7750     10095#   10175
TSTOFF       37      6815     10108#
TSTPC        36      1474      1825      1838      6896      6962     10097#
TSTPNO     7002      7005#
TSTPNT       31      1035      6995#
TSTPNX     7007      7010#
TSTRAO     1662#     1668
TSTRAA     1647      1672#
TSTRAN      600      1647#
TSTREL     1815      1819     10109#
TSTREP       37      1603      1649      1687      1783      1890      1893      1915     10102#
TSTREQ     1609      1619      1651      1688      1689      1698      1730      1741      1789     10103#
TSTS1        80#     2004      2245
TSTS10       80#     2011      2252
TSTS11       81#     2012      2253
TSTS12       81#     2013      2254
TSTS13       81#     2014      2255
TSTS14       81#     2015      2256
TSTS15       81#     2016      2257
TSTS16       81#     2017      2258
TSTS17       81#     2018      2259
TSTS2        80#     2005      2246
TSTS20       81#     2019      2260
```

```
TSTS21        82#     2020      2261
TSTS22        82#     2021      2262
TSTS23        82#     2022      2263
TSTS24        82#     2023      2264
TSTS25        82#     2024      2265
TSTS26        82#     2025      2266
TSTS27        82#     2026      2267
TSTS3         80#     2006      2247
TSTS30        82#     2027      2268
TSTS31        83#     2028      2269
TSTS32        83#     2029      2270
TSTS33        83#     2030      2271
TSTS34        83#     2031      2272
TSTS35        83#     2032      2273
TSTS36        83#     2033      2274
TSTS37        83#     2034      2275
TSTS4         80#     2007      2248
TSTS40        83#     2035      2276
TSTS41        84#     2036      2277
TSTS42        84#     2037      2278
TSTS43        84#     2038      2279
TSTS44        84#     2039      2280
TSTS45        84#     2040      2281
TSTS46        84#     2041      2282
TSTS47        84#     2042      2283
TSTS5         80#     2008      2249
TSTS50        84#     2043      2284
TSTS51        85#     2044      2285
TSTS52        85#     2045      2286
TSTS53        85#     2046      2287
TSTS54        85#     2047      2288
TSTS55        85#     2048      2289
TSTS56        85#     2049      2290
TSTS57        85#     2050      2291
TSTS6         80#     2009      2250
TSTS60        85#     2051      2292
TSTS61        86#     2052      2293
TSTS62        86#     2053      2294
TSTS63        86#     2054      2295
TSTS64        86#     2055      2296
TSTS65        86#     2056      2297
TSTS66        86#     2057      2298
TSTS67        86#     2058      2299
TSTS7         80#     2010      2251
TSTS70        86#     2059      2300
TSTSAV       413#      413       421       453
TSTSEO      1201#
TSTSEG       244      1200      1274#
TSTSEL       606       613      1197#     3111      3120
TSTSEX       227      1241      1270      1288      1304#
TSTSG0      1277      1286#
TSTSP0      1613      1620#
TSTSPC       434      1601#
```

```
TSTSSW       636       637       638       639       640       641       642       643       644       645       646       647       648       649
TSTSSW     636    637    638    639    640    641    642    643    644    645    646    647    648    649        SEQ 0457
           650    651    652    653    654    655    656    657    658    659    660   1534#
TSTSUB      36   6841   6894   7001   7935   8387   8780   9176   9247   9764  10096#
TSTSWI     631   1517#
TSTSWP     632   1523#
TSTTAO    1557   1563   1569#
TSTTAK     602   1553#  3106
TSTTAN     603   1551#  3107
TSTU1       94#  2100   2226
TSTU10      94#  2107   2379
TSTU11      95#  2108   2380
TSTU12      95#  2109   2381
TSTU13      95#  2110   2382
TSTU14      95#  2111   2415
TSTU15      95#  2112   2416
TSTU16      95#  2113   2417
TSTU17      95#  2114   2418
TSTU2       94#  2101   2227
TSTU20      95#  2115   2419
TSTU21      96#  2116   2420
TSTU22      96#  2117   2421
TSTU23      96#  2118   2422
TSTU24      96#  2119   2423
TSTU25      96#  2120   2424
TSTU26      96#  2121   2425
TSTU27      96#  2122   2426
TSTU3       94#  2102   2228
TSTU30      96#  2123   2427
TSTU31      97#  2124   2428
TSTU32      97#  2125   2429
TSTU33      97#  2126   2430
TSTU34      97#  2127   2386
TSTU35      97#  2128   2387
TSTU36      97#  2129   2388
TSTU37      97#  2130   2389
TSTU4       94#  2103   2229
TSTU40      97#  2131   2393
TSTU41      98#  2132   2241
TSTU5       94#  2104   2230
TSTU6       94#  2105   2239
TSTU7       94#  2106   2240
TTIME     8841#
TTNBRF    1739
TTPN0     8863   8865   8873#
TTPN1     8869   8874   8881#
TTPN2     8882   8889#
TTPNT       52   8854#
TTPNX     8891   8907#
TTYYES     120#  1279   1290
TUSER     6833   6850
TWORD       32   6622   6627  10130#
TXALL     7493   7494
TXTINH    7006   7506   8053   8234   8243   8263   8521   8881   8892   8894   8898   8900   9290   9298
```

```
          9315   9317   9321   9323   9331   9899   9907   9927  10004  10006  10010  10012  10026  10028
```

```
           9315      9317      9321      9323      9331      9899      9907      9927     10004     10006     10010     10012     10026     10028
          10032     10034     10048     10050     10054     10056
UDEBUG       15       139#      139       205       211       352       359      1103      2837      2844      7641      7694
UNAME       121#     3640      3642      3647      3650
USER        258      1785      1878      7017      7643      7695      7941      8394      8712      9107      9695
USRION      115#     1100
UUT          67      1798      1824     10086#
WRDERR      124#     6718
XWORDL     6344#     6344      6349#     6349      6379
XWORDR     6346#     6346      6352#     6352      6380
XXW         400#      400      2879#     2879      4639#     4639
XXX        1676#     1676      3101#     3101      6873
ZAL        1932#
ZCB        1934#
ZEB        1930#
ZMP        1933#
ZSE        1931#
$$TOGG      184     10195
$ARG2      5262#     5262      5267#     5267      5268#     5268      5269#     5269      5270#     5270      5271#     5271      5276#     5276
           5545#     5545      5546#     5546      5551#     5551      5552#     5552      5553#     5553      5554#     5554      5555#     5555
           5560#     5560      5979#     5979      6551#     6551      6552#     6552      6553#     6553      6554#     6554
$B         5262#     5262      5267#     5267      5268#     5268      5269#     5269      5270#     5270      5271#     5271      5276#     5276
           5545#     5545      5546#     5546      5551#     5551      5552#     5552      5553#     5553      5554#     5554      5555#     5555
           5560#     5560      5979#     5979      6551#     6551      6552#     6552      6553#     6553      6554#     6554
$CHR       5262#     5262      5267#     5267      5268#     5268      5269#     5269      5270#     5270      5271#     5271      5276#     5276
           5545#     5545      5546#     5546      5551#     5551      5552#     5552      5553#     5553      5554#     5554      5555#     5555
           5560#     5560      5979#     5979      6551#     6551      6552#     6552      6553#     6553      6554#     6554
$GARG      5262#     5262      5267#     5267      5268#     5268      5269#     5269      5270#     5270      5271#     5271      5276#     5276
           5545#     5545      5546#     5546      5551#     5551      5552#     5552      5553#     5553      5554#     5554      5555#     5555
           5560#     5560      5979#     5979      6551#     6551      6552#     6552      6553#     6553      6554#     6554
$ONETM      134       151       168       170      6975     10199
$TTCHR      403       409       415      2882      2890      2896      4602      4616      4642      4658      4686
$TWCNT      401      2880      4640
%ADDR      5262#     5262      5267#     5267      5268      5268#     5269      5269#     5270      5270#     5271      5271#     5276#     5276
           5545#     5545      5546      5546#     5551#     5551      5552      5552#     5553      5553#     5554      5554#     5555      5555#
           5560#     5560      5979#     5979      6551#     6551      6552      6552#     6553      6553#     6554      6554#
%ML        5262#     5262      5267#     5267      5268#     5268      5269#     5269      5270#     5270      5271#     5271      5276#     5276
           5545#     5545      5546#     5546      5551#     5551      5552#     5552      5553#     5553      5554#     5554      5555#     5555
           5560#     5560      5979#     5979      6551#     6551      6552#     6552      6553#     6553      6554#     6554
%MR        5262#     5262      5267#     5267      5268#     5268      5269#     5269      5270#     5270      5271#     5271      5276#     5276
           5545#     5545      5546#     5546      5551#     5551      5552#     5552      5553#     5553      5554#     5554      5555#     5555
           5560#     5560      5979#     5979      6551#     6551      6552#     6552      6553#     6553      6554#     6554
.CGOT       123#     1537      6731
.CLOSE      120#     1077
.COMM       123#      423       455      1205      1246      1367      2579      2625      2904      3265      3433      4672
.CONI       112#
.CONO       112#
.DARG       123#      670       998      1011      1615      1694      3562      4080      4154      5103      5187      5195      5298      5306
           5394      5473      5481      5820      6654
.DATAI      112#
.DATAO      112#     5155      5441
.DBCMD      454      2903      2943#
.DBDIS      462      2908      3096#     3221
```

```
.HLCMD     3264    3278#
.INPNT      115#   2829
.INWD1      115#    386
.ISWT       122#   1520    3213
.JBSA     10198
.LADDR     5262    5267    5276    5545    5551    5560    5979    6551
.LAND      5276    5560
.LCCER     5268    5552
.LCENA     5268    5552    6552
.LCJP      5268    5552    6552
.LCONT     5270    5554
.LCRTN     5979
.LD        5262    5267    5268    5269    5270    5271    5276    5545    5546    5551    5552    5553    5554    5555
           5560    5979    6551    6552    6553    6554
.LJ        5262    5267    5268    5269    5271    5276    5545    5546    5551    5552    5553    5555    5560    6551
           6552    6553    6554
.LJMAP     5262    5267    5269    5271    5276    5545    5546    5551    5553    5555    5560    6551    6553    6554
.LLDLM     5555    5560
.LMGC      5262    5270    5546    5554    5555
.LOENA     5262    5546    5555    5560
.LOR       5262    5270    5545    5546    5554    5555
.LRDLM     5545
.LSOA      5262    5276    5546    5555    5560
.LSDO      5270    5545    5554
.LSELE     5262    5270    5546    5554
.LSWIT      122#
.MAND      5276    5560
.MCCER     5268    5552
.MCENA     5268    5552    6552
.MCJP      5268    5552    6552
.MCONT     5270    5554
.MCRTN     5979
.MD        5262    5267    5268    5269    5270    5271    5276    5545    5546    5551    5552    5553    5554    5555
           5560    5979    6551    6552    6553    6554
.MJ        5262    5267    5268    5269    5271    5276    5545    5546    5551    5552    5553    5555    5560    6551
           6552    6553    6554
.MJMAP     5262    5267    5269    5271    5276    5545    5546    5551    5553    5555    5560    6551    6553    6554
.MLDLM     5555    5560
.MMGC      5262    5270    5546    5554    5555
.MOENA     5262    5546    5555    5560
.MOR       5262    5270    5545    5546    5554    5555
.MRDLM     5545
.MSOA      5262    5276    5546    5555    5560
.MSDO      5270    5545    5554
.MSELE     5262    5270    5546    5554
.OARG       123#   3470    3496    3587    3600    3681    3722    3732    3825    3865    3895    3911    3989    4300
           4341    4426    4436    4521    4529    4537    4545    4553    5008    5016    5118    5408    5582    5625
           5670    5680    5743    5753    6000    6047    6088    6098    6156    6166    6454    6562    6589
.PIOFF      112#   9770
.PION       112#   9851
.PNTOC      122#
.RAND      5276    5560
.RCCER     5268    5552
```

```
.RCENA    5268   5552   6552
.RCJP     5268   5552   6552
.RCONT    5270   5554
.RCRTN    5979
.RD       5262   5267   5268   5269   5270   5271   5276   5545   5546   5551   5552   5553   5554   5555
          5560   5979   6551   6552   6553   6554
.RESET     121#   190
.RJ       5262   5267   5268   5269   5271   5276   5545   5546   5551   5552   5553   5555   5560   6551
          6552   6553   6554
.RJMAP    5262   5267   5269   5271   5276   5545   5546   5551   5553   5555   5560   6551   6553   6554
.RLDLM    5555   5560
.RMGC     5262   5270   5546   5554   5555
.ROENA    5262   5546   5555   5560
.ROR      5262   5270   5545   5546   5554   5555
.RRDLM    5545
.RSOA     5262   5276   5546   5555   5560
.RSDO     5270   5545   5554
.RSELE    5262   5270   5546   5554
.RSWIT     122#   176
.SARG      123#   938   1198   1239   1360   1456   1555   1561   3258   3426   3515   3539   3630   3636
          6234   6269
.STCMD    3432   3440#
.STDIS    3437   3453#
.SWCHP     122#  1526   3218
```

```
ALTCHK     400     1738     2879     4639
CALC      5262     5267     5268     5269     5270     5271     5276     5545     5546     5551     5552     5553     5554     5555          SEQ 0461
          5560     5979     6551     6552     6553     6554
CONCAT    5262     5267     5268     5269     5270     5271     5276     5545     5546     5551     5552     5553     5554     5555
          5560     5979     6551     6552     6553     6554
DISMS     7042
ENDUUO     262
EOPUUO     268
FIELD     5262     5267     5268     5269     5270     5271     5276     5545     5546     5551     5552     5553     5554     5555
          5560     5979     6551     6552     6553     6554
FMSG       393      444     1892     1895     2871     4517     4704     4705     4706     4707     4708     4709     4710     4711
          4712     4713     4714     4715     4716     4717     4718     4719     4720     4721
FMSGC      440      946     1003     1018     1278     1289     1464     1572     1580     1672     1736     1889     4509     6244
          6279
FMSGCD     326      467      608      669      937      943      997     1010     1047     1058     1061     1073     1197     1206
          1223     1231     1238     1247     1258     1266     1368     1442     1446     1449     1461     1487     1517     1523
          1534     1554     1560     1614     1693     2813     2824     2887     2905     3098     3115     3192     3210     3215
          3266     3428     3434     3469     3477     3495     3514     3538     3561     3568     3586     3593     3599     3602
          3609     3629     3635     3661     3677     3721     3731     3741     3748     3766     3810     3824     3827     3839
          3851     3864     3867     3881     3894     3897     3910     3939     3950     3960     3988     4002     4005     4010
          4026     4030     4033     4041     4071     4076     4096     4099     4153     4175     4247     4251     4254     4259
          4270     4275     4293     4299     4302     4307     4319     4340     4343     4358     4425     4435     4446     4452
          4456     4520     4523     4528     4531     4536     4539     4544     4547     4552     4555     4653     4673     4825
          4826     4827     4828     4841     4842     4843     4844     4863     4870     4871     4872     5007     5015     5026
          5032     5036     5102     5105     5110     5117     5120     5186     5194     5205     5210     5214     5297     5305
          5316     5322     5326     5393     5396     5402     5407     5410     5472     5480     5491     5496     5500     5575
          5581     5587     5618     5624     5630     5669     5679     5690     5695     5699     5742     5752     5763     5768
          5772     5819     5905     5933     5955     5958     5993     5999     6005     6040     6046     6052     6087     6097
          6108     6113     6117     6155     6165     6176     6181     6185     6233     6236     6241     6252     6268     6271
          6276     6287     6297     6333     6395     6424     6453     6456     6468     6561     6564     6588     6591     6612
          6639     6653     6682     6693     6707     6713     6727     6741     6797
FMSGD     1356     1578     1586     1675     1896     3253     3422     4600     4829     4833     4837     4845     4859     4864
FSELEC    1571
GET        400      829      830     1159     1160     1190     1191     1335     1349     1640     1641     1676     1923     1924
          2513     2519     2555     2617     2618     2879     3101     4211     4235     4236     4639     4970     4971     4991
          4992     5608     5651     5721     6026     6030     6069     6134     6309     6387     6388     6734     6750     6751
          6866     6867     6872     6873     6911     6912     6943     6944     6988     6989     7010     7011     7031     7032
          7046     7047     7073     7074     7104     7198     7199     7244     7245     7289     7290     7372     7373     7396
          7397     7445     7446     7480     7484     7511     7512     7543     7544     7607     7608     7627     7628     7629
          7672     7673     7706     7708     7795     7796     7829     7830     7922     7923     7995     7996     8007     8008
          8021     8022     8037     8038     8091     8092     8195     8196     8278     8279     8373     8374     8464     8465
          8476     8477     8490     8491     8506     8507     8587     8588     8691     8692     8722     8723     8735     8736
          8749     8750     8765     8766     8775     8776     8833     8834     8907     8908     9016     9017     9067     9068
          9119     9120     9131     9132     9145     9146     9161     9162     9171     9172     9242     9243     9260     9261
          9369     9370     9516     9517     9550     9551     9584     9585     9618     9619     9653     9654     9678     9679
          9707     9708     9719     9720     9733     9734     9749     9750     9759     9760     9856     9857    10063    10064
GETJI     7827
GO         190      196      201      209      224      225      227      234      239      244      248      252      253      254
           257      264      267      324      330      336      346      357      374      375      391      396      423      430
           434      448      455      457      458      460      462      468      473      478      605      606      609      611
           670      726      731      735      796      938      942      944      950      955      964      998     1007     1011
          1022     1034     1035     1048     1050     1059     1062     1063     1064     1074     1076     1077     1099     1100
          1105     1113     1121     1130     1138     1158     1198     1205     1207     1220     1228     1239     1246     1248
          1255    1263    1279    1290    1360    1362    1367    1369    1450    1456    1460    1462    1468    1473
          1488    1494    1502    1518    1520    1524    1526    1535    1539    1555    1561    1569    1604    1615
          1656    1663    1666    1694    1703    1767    1769    1782    1784    1799    1811    1821    1825    1830
          1848    1857    1862    1880    1897    1899    1900    1910    1912    1921    2501    2579    2587    2625
          2814    2820    2821    2823    2829    2842    2859    2862    2869    2874    2888    2904    2906    2908
          3099    3110    3111    3116    3118    3120    3193    3211    3213    3216    3218    3258    3260    3265
          3267    3426    3429    3433    3435    3437    3470    3478    3496    3515    3518    3539    3542    3562
          3569    3587    3594    3600    3603    3607    3630    3636    3659    3664    3665    3666    3667    3668
          3669    3670    3681    3688    3722    3732    3742    3749    3767    3811    3813    3815    3825    3828
          3831    3840    3843    3852    3854    3865    3868    3871    3882    3884    3895    3898    3901    3911
          3914    3940    3942    3951    3953    3961    3967    3968    3969    3989    3995    4001    4003    4006
          4011    4013    4027    4029    4031    4034    4043    4080    4083    4095    4097    4100    4102    4111
          4117    4154    4160    4176    4248    4250    4252    4255    4261    4271    4273    4274    4276    4281
          4294    4300    4303    4305    4306    4308    4311    4320    4322    4324    4325    4341    4344    4346
          4348    4350    4359    4361    4363    4364    4426    4436    4448    4453    4457    4473    4505    4521
          4524    4529    4532    4537    4540    4545    4548    4553    4556    4565    4584    4594    4601    4622
          4627    4633    4654    4656    4664    4672    4674    4676    4685    4692    4882    4902    4906    4929
          4936    4955    4960    4962    4967    4969    4981    4982    4988    4989    5008    5016    5028    5033
          5037    5049    5067    5071    5073    5074    5093    5103    5106    5111    5118    5121    5131    5142
          5151    5153    5155    5157    5170    5187    5195    5206    5211    5215    5227    5238    5244    5246
          5257    5298    5306    5318    5323    5327    5339    5349    5357    5361    5363    5365    5366    5384
          5394    5397    5403    5408    5411    5421    5430    5437    5439    5441    5443    5456    5473    5481
          5492    5497    5501    5513    5520    5524    5526    5528    5540    5576    5582    5588    5593    5600
          5607    5619    5625    5631    5636    5643    5650    5670    5680    5691    5696    5700    5705    5712
          5720    5743    5753    5764    5769    5773    5778    5787    5796    5820    5835    5906    5934    5937
          5939    5944    5954    5956    5959    5974    5994    6000    6006    6011    6018    6023    6029    6041
          6047    6053    6058    6065    6068    6088    6098    6109    6114    6118    6123    6130    6133    6156
          6166    6177    6182    6186    6191    6200    6204    6212    6234    6237    6240    6242    6248    6250
          6253    6255    6256    6269    6272    6275    6277    6283    6285    6288    6290    6298    6301    6303
          6304    6308    6323    6334    6338    6340    6341    6347    6353    6381    6385    6396    6398    6401
          6410    6411    6425    6427    6429    6438    6439    6454    6457    6460    6469    6475    6485    6489
          6491    6493    6494    6525    6562    6565    6567    6578    6589    6592    6594    6602    6613    6623
          6625    6629    6640    6654    6683    6685    6694    6697    6698    6708    6714    6717    6728    6733
          6742    6747    6798    6800    6801    6816    6823    6935    6936    6976    7005    7024    7028    7058
          7062    7069    7102    7106    7122    7123    7141    7148    7151    7153    7161    7164    7167    7196
          7227    7231    7237    7280    7281    7369    7472    7473    7505    7510    7565    7586    7598    7602
          7616    7624    7652    7658    7714    7753    7779    7945    7956    7961    7965    7974    8002    8015
          8020    8029    8034    8141    8142    8147    8149    8154    8158    8168    8169    8172    8173    8179
          8180    8185    8187    8235    8244    8399    8409    8414    8418    8426    8427    8436    8471    8484
          8489    8498    8503    8705    8709    8716    8718    8730    8743    8748    8757    8762    8771    8784
          8785    8800    8801    8811    8812    8819    9031    9033    9043    9045    9052    9054    9097    9104
          9112    9114    9126    9139    9144    9153    9158    9167    9180    9181    9188    9189    9203    9204
          9213    9228    9251    9291    9334    9365    9539    9573    9607    9642    9658    9659    9673    9683
          9692    9700    9702    9714    9727    9732    9741    9746    9755    9766    9767    9770    9775    9776
          9786    9791    9844    9851    9863    9865    9867    9868    9900    9941   10153   10155   10160   10162
         10168   10170
GTAD      7660    7774
HALT      1080
HIBER     7044
MFLD      5262#   5262    5267#   5267    5268#   5268    5269#   5269    5270#   5270    5271#   5271    5276#   5276
          5545#   5545    5546#   5546    5551#   5551    5552#   5552    5553#   5553    5554#   5554    5555#   5555
          5560#   5560    5979#   5979    6551#   6551    6552#   6552    6553#   6553    6554#   6554
MWORD     5262    5267    5268    5269    5270    5271    5276    5545    5546    5551    5552    5553    5554    5555
```

```
          5560   5979   6551   6552   6553   6554                                                                        SEQ 0463
PCRL       795    828   1030   1039   1189   1314   1495   1503   1864   1922   2860   3483   3487   3503
          3507   3525   3549   3574   3612   3621   3645   3652   3701   3708   3714   3751   3816   4014
          4044   4123   4144   4210   4285   4899   4917   4931   4944   5091   5382   5810   5972   6226
          6386   6405   6417   6433   6445   6461   6520   6674   6686   6699   6719   6787   6835   7222
          7270   7288   7534   7921   8372   8690   9015   9515  10182
PCRL2     3759
PCRLF      332    392    447    949   1006   1021   1467   2870   4262   4511   4592   4595   4646   4651
          4655   4675   6247   6282
PFORCE     391    485    680    700   1371   2869   2914   3269   3769
PGMINT     191
PJRST      703   1495   3698   3704   5862   5869   5876   5911   5919   5927
PNT1      7304   7307   7312   7315   7318   7321   7324   7336   7339   7360   7363   7366   7567   7590
PNT4      4039   4137   4140   4200   4203   6377   7223   7384   7386   7391   7393   7395   8060   8067
          8070   8077   8084   8087   8232   8240   8246   8250   8267   8528   8536   8540   8543   8559
          8567   8571   8574   8858   8868   8877   9293   9296   9302   9305   9902   9905   9911   9914
PNT4F     4516   4599
PNTAL     2511
PNTCIF    1575   1583
PNTDCF    1674   1894
PNTDEC    1917   3577   6260   6318   6362   6365   6668   6773   6776   7217   7976   8063   8080   8438
          8531   8562
PNTHW     3620   4143   4206   4284   5076   5368   6671   6780   6783   6786   7504   8237   8450   8549
          8552   8580   8583   8821   8885   8897   8903   9230   9308   9320   9326   9541   9575   9609
          9644   9675   9846   9917   9920   9926  10009  10015  10031  10037  10053  10059  10181
PNTMSF     326    393    440    444    467    608    669    937    943    946    997   1003   1010   1018
          1047   1058   1061   1073   1197   1206   1223   1231   1238   1247   1258   1266   1278   1289
          1359   1368   1442   1446   1449   1461   1464   1487   1517   1523   1534   1554   1560   1572
          1578   1580   1586   1614   1672   1675   1693   1736   1889   1892   1895   1896   2813   2824
          2871   2887   2905   3098   3115   3192   3210   3215   3257   3266   3425   3428   3434   3469
          3477   3495   3514   3538   3561   3568   3586   3593   3599   3602   3609   3629   3635   3661
          3680   3721   3731   3741   3748   3766   3810   3824   3827   3839   3851   3864   3867   3881
          3894   3897   3910   3939   3950   3960   3988   4002   4005   4010   4026   4030   4033   4041
          4071   4076   4096   4099   4153   4175   4247   4251   4254   4259   4270   4275   4293   4299
          4302   4307   4319   4340   4343   4358   4425   4435   4446   4452   4456   4509   4517   4520
          4523   4528   4531   4536   4539   4544   4547   4552   4555   4589   4600   4653   4673   4704
          4705   4706   4707   4708   4709   4710   4711   4712   4713   4714   4715   4716   4717   4718
          4719   4720   4721   4825   4826   4827   4828   4832   4836   4840   4841   4842   4843   4844
          4858   4862   4863   4869   4870   4871   4872   5007   5015   5026   5032   5036   5102   5105
          5110   5117   5120   5186   5194   5205   5210   5214   5297   5305   5316   5322   5326   5393
          5396   5402   5407   5410   5472   5480   5491   5496   5500   5575   5581   5587   5618   5624
          5630   5669   5679   5690   5695   5699   5742   5752   5763   5768   5772   5819   5905   5933
          5955   5958   5993   5999   6005   6040   6046   6052   6087   6097   6108   6113   6117   6155
          6165   6176   6181   6185   6233   6236   6241   6244   6252   6268   6271   6276   6279   6287
          6297   6333   6395   6424   6453   6456   6468   6561   6564   6588   6591   6612   6639   6653
          6682   6693   6707   6713   6727   6741   6797
PNTMSG     486    688    694    699    800    802    804    806    807    814    817    821   1167   1168
          1169   1171   1172   1173   1174   1175   1176   1177   1178   1179   1181   1182   1183   1184
          1185   1186   1187   1188   1306   1309   1313   1344   1346   1348   1388   1398   1410   1415
          1423   1433   1706   1707   1708   1848   1854   1855   1861   1911   1918   2915   3298   3313
          3318   3323   3327   3337   3342   3347   3352   3360   3365   3370   3377   3383   3389   3397
          3405   3415   3484   3504   3526   3528   3530   3550   3552   3554   3575   3578   3613   3618
          3646   3649   3663   3687   3693   3698   3704   3710   3770   3814   4037   4087   4103   4106
```

```
           4109   4126   4129   4135   4138   4141   4159   4201   4204   4207   4282   4326   4329   4333
           4901   4926   4933   5053   5058   5061   5347   5349   5352   5780   5797   5945   5946   5962
           5963   5964   5965   5966   6193   6213   6258   6261   6316   6319   6320   6326   6360   6363
           6366   6375   6378   6382   6400   6407   6435   6503   6512   6530   6531   6532   6533   6534
           6535   6536   6537   6538   6539   6540   6541   6542   6543   6544   6545   6580   6605   6663
           6666   6669   6759   6761   6762   6763   6764   6765   6766   6767   6768   6771   6774   6777
           6778   6781   6784   6789   6822   6824   6832   6834   7003   7008   7009   7216   7218   7219
           7224   7228   7232   7243   7261   7273   7302   7305   7310   7313   7316   7319   7334   7337
           7342   7343   7344   7345   7346   7347   7348   7349   7350   7351   7352   7353   7354   7355
           7356   7357   7358   7502   7536   7550   7551   7552   7553   7554   7555   7556   7557   7558
           7559   7560   7561   7562   7563   7564   7565   7571   7572   7573   7574   7575   7576   7577
           7578   7579   7580   7581   7582   7583   7584   7585   7589   7919   7974   7977   7980   7983
           8002   8015   8029   8058   8061   8064   8065   8068   8075   8078   8081   8082   8085   8230
           8235   8238   8241   8244   8247   8256   8265   8268   8274   8370   8436   8439   8442   8445
           8448   8471   8484   8498   8526   8529   8532   8534   8538   8541   8547   8550   8557   8560
           8563   8565   8569   8572   8578   8581   8688   8718   8730   8743   8757   8771   8819   8822
           8825   8856   8866   8875   8883   8893   8895   8899   8901   9013   9054   9056   9057   9058
           9059   9062   9114   9126   9139   9153   9167   9228   9231   9234   9291   9294   9300   9303
           9306   9316   9318   9322   9324   9338   9340   9342   9344   9346   9348   9350   9352   9354
           9356   9513   9539   9542   9545   9573   9576   9579   9607   9610   9613   9642   9645   9648
           9673   9702   9714   9727   9741   9755   9844   9847   9900   9903   9909   9912   9915   9918
           9924   9929   9931   9932   9933   9934   9935   9936   9938   9939   9940   9943   9945   9947
           9948   9949   9950   9951   9954   9957   9965   9968   9970   9971   9972   9973   9974   9975
           9976   9985   9988   9990   9991  10001  10005  10007  10011  10013  10023  10027  10029  10033
          10035  10045  10049  10051  10055  10057  10178
PNTOCC     4900   4932   5799   5967   6215   6406   6434   7272   7301   7309   7327   7330   7333
PNTOCF      446    948   1005   1020   1466   4634   6246   6281
PNTOCS      819   1509   1853   3486   3506   3695   3700   3707   3713   4108   4128   4209   4328   4332
           5060   5351   6325   6770   6829   7000   7004   7252   7979   7982   7985   8004   8017   8031
           8441   8444   8447   8473   8486   8500   8720   8732   8745   8759   8824   8827   9061   9064
           9116   9128   9141   9155   9233   9236   9544   9547   9578   9581   9612   9615   9647   9650
           9704   9716   9729   9743   9849   9953   9956   9959   9980   9993  10003  10025  10047
PNTSIX     1507   1851   3648   3651   6827   6998
PNTSXF      443   1574   1577   1582   1585
PSP         820   1510   2512   5800   6216   7322   7325   7328   7331   7340   7361   7364
PSPF       4635
PUT         752    753   1098   1099   1166   1167   1322   1338   1628   1629   1909   1910   2495   2501
           2526   2571   2572   4198   4219   4220   4951   4952   4977   4978   5605   5648   5718   6019
           6027   6066   6131   6306   6371   6372   6730   6744   6745   6814   6815   6893   6894   6927
           6928   6961   6962   6995   6996   7022   7023   7036   7037   7056   7057   7101   7114   7115
           7213   7214   7260   7261   7299   7300   7382   7383   7416   7417   7471   7501   7502   7522
           7523   7597   7598   7615   7616   7617   7651   7652   7702   7713   7714   7823   7824   7895
           7896   8019   8020   8033   8034   8051   8052   8123   8124   8229   8230   8345   8346   8488
           8502   8519   8520   8664   8665   8747   8748   8761   8762   8854   8855   8986   8987   9143
           9144   9157   9158   9285   9286   9485   9486   9731   9732   9745   9746   9894   9895  10157
          10158  10164  10165  10172  10173
RGET        829   1159   1190   1640   1923   2513   2617   4235   4970   4991   6387   6750   6866   6672
           6911   6943   6988   7010   7031   7046   7073   7198   7244   7289   7372   7396   7445   7511
           7543   7607   7628   7672   7795   7829   7922   7995   8007   8021   8037   8091   8195   8278
           8373   8464   8476   8490   8506   8587   8691   8722   8735   8749   8765   8775   8833   8907
           9016   9067   9119   9131   9145   9161   9171   9242   9260   9369   9516   9550   9584   9618
           9653   9678   9707   9719   9733   9749   9759   9856  10063
RPUT        752   1098   1166   1628   1909   2501   2571   4219   4951   4977   6371   6744   6814   6893
```

```
             6927    6961    6995    7022    7036    7056    7114    7213    7260    7299    7382    7416    7501    7522
             7597    7615    7651    7713    7823    7895    8019    8033    8051    8123    8229    8345    8519    8664
             8747    8761    8854    8986    9143    9157    9285    9485    9731    9745    9894   10157   10164   10172
RTN           237     242     337     394     481     592     607     610     612     671     736     748     750     830
              939     940     945     951     965     999    1008    1012    1023    1032    1041    1049    1060    1066
             1075    1160    1191    1199    1208    1224    1226    1232    1234    1240    1249    1259    1261    1267
             1269    1315    1336    1350    1361    1364    1370    1374    1378    1451    1457    1458    1463    1469
             1489    1496    1504    1519    1521    1525    1527    1536    1540    1556    1562    1579    1588    1616
             1641    1677    1695    1788    1791    1856    1903    1908    1924    2494    2514    2520    2556    2618
             2815    2822    2825    2872    3096    3100    3102    3114    3117    3119    3123    3194    3200    3212
             3214    3217    3219    3259    3262    3268    3272    3276    3427    3430    3436    3438    3471    3479
             3481    3488    3497    3501    3508    3516    3519    3521    3523    3531    3540    3543    3545    3547
             3555    3563    3570    3572    3579    3588    3595    3601    3604    3608    3610    3622    3631    3637
             3643    3653    3660    3662    3671    3682    3689    3691    3702    3709    3715    3723    3733    3743
             3750    3760    3768    3771    3812    3817    3826    3829    3832    3841    3844    3853    3858    3866
             3869    3874    3883    3887    3896    3899    3904    3912    3932    3941    3943    3952    3954    3962
             3970    3990    4004    4007    4009    4015    4028    4032    4035    4045    4081    4098    4101    4125
             4145    4155    4163    4168    4177    4181    4212    4236    4249    4253    4256    4258    4263    4272
             4277    4286    4295    4301    4304    4309    4312    4321    4334    4342    4345    4351    4360    4365
             4427    4437    4449    4454    4458    4506    4522    4525    4530    4533    4538    4541    4546    4549
             4554    4557    4570    4596    4696    4726    4728    4730    4732    4734    4736    4738    4740    4742
             4744    4746    4748    4750    4752    4754    4756    4758    4760    4765    4767    4769    4771    4773
             4775    4777    4779    4781    4783    4785    4787    4789    4791    4793    4795    4797    4799    4885
             4890    4918    4945    4971    4992    5009    5017    5029    5034    5038    5094    5104    5107    5112
             5119    5122    5171    5188    5196    5207    5212    5216    5258    5299    5307    5319    5324    5328
             5353    5385    5395    5398    5404    5409    5412    5457    5474    5482    5493    5498    5502    5541
             5577    5583    5584    5589    5620    5626    5627    5632    5671    5681    5692    5697    5701    5725
             5744    5754    5765    5770    5774    5798    5811    5821    5838    5907    5935    5975    5995    6001
             6002    6007    6042    6048    6049    6054    6089    6099    6110    6115    6119    6138    6157    6167
             6178    6183    6187    6214    6227    6235    6238    6243    6249    6254    6262    6270    6273    6278
             6284    6289    6291    6299    6302    6327    6335    6339    6367    6388    6397    6399    6418    6426
             6428    6446    6455    6458    6462    6470    6526    6563    6566    6581    6590    6593    6606    6614
             6630    6641    6655    6675    6684    6687    6695    6700    6709    6715    6720    6729    6735    6743
             6751    6790    6799    6802    6868    6873    6912    6944    6989    7011    7032    7047    7074    7107
             7109    7113    7155    7199    7212    7245    7254    7290    7371    7373    7397    7446    7482    7491
             7512    7544    7568    7588    7591    7608    7629    7642    7645    7647    7673    7696    7698    7707
             7709    7796    7830    7923    7986    7996    8005    8009    8018    8023    8032    8038    8092    8197
             8199    8201    8203    8205    8242    8248    8279    8374    8451    8465    8474    8478    8487    8492
             8501    8507    8588    8692    8721    8724    8733    8737    8746    8751    8760    8766    8772    8776
             8828    8834    8908    9017    9065    9068    9117    9120    9129    9133    9142    9147    9156    9162
             9168    9172    9237    9243    9261    9297    9370    9517    9548    9551    9582    9585    9616    9619
             9651    9654    9676    9679    9705    9708    9717    9721    9730    9735    9744    9750    9756    9760
             9789    9794    9850    9857    9869    9906    9946   10064   10184
SCOPER       7486
SYERR        7667    7787
TMSG          817    1171    1172    1173    1174    1175    1176    1177    1178    1181    1182    1183    1184    1185
             1186    1187    1188    1854    1911    1918    3484    3504    3526    3550    3575    3613    3618    3646
             3649    3698    3704    3710    4126    4138    4141    4201    4204    4207    4329    4901    4933    5058
             5061    5347    5352    6363    6407    6435    6530    6531    6532    6533    6534    6535    6536    6537
             6538    6539    6540    6541    6542    6543    6544    6545    6663    6669    6771    6774    6777    6781
             6784    6822    6824    6832    6834    7003    7008    7218    7224    7228    7232    7273    7302    7305
             7310    7313    7316    7319    7334    7337    7342    7343    7344    7345    7346    7347    7348    7349
             7350    7351    7352    7353    7354    7355    7356    7357    7358    7536    7550    7551    7552    7553
```

```
              7554      7555      7556      7557      7558      7559      7560      7561      7562      7563      7564      7565      7571      7572
              7573      7574      7575      7576      7577      7578      7579      7580      7581      7582      7583      7584      7585      7589
              7977      7980      7983      8061      8064      8078      8081      8268      8439      8442      8445      8448      8529      8532
              8560      8563      8822      8825      9056      9057      9058      9059      9062      9231      9234      9294      9340      9342
              9344      9346      9348      9350      9352      9354      9356      9542      9545      9576      9579      9610      9613      9645
              9648      9847      9903      9929      9931      9932      9933      9934      9935      9936      9938      9939      9940      9943
              9945      9947      9948      9949      9950      9954      9957      9970      9971      9972      9973      9974      9975      9976
              9991
TMSGC          681       689       814      1167      1168      1169      1179      1306      1309      1313      1848      1855      1861      3693
              3814      4037      4103      4106      4135      4282      4326      4926      5053      5349      5780      5797      5945      5946
              5962      5963      5964      5965      5966      6193      6213      6258      6316      6320      6360      6375      6378      6382
              6400      6503      6512      6666      6759      6761      6762      6763      6764      6765      6766      6767      6768      6778
              7216      7219      7243      7261      7502      7974      8002      8015      8029      8058      8065      8068      8075      8082
              8085      8230      8235      8238      8241      8244      8247      8256      8265      8274      8436      8471      8484      8498
              8526      8534      8538      8541      8547      8550      8557      8565      8569      8572      8578      8581      8718      8730
              8743      8757      8771      8819      8856      8866      8875      8883      8893      8895      8899      8901      9054      9114
              9126      9139      9153      9167      9228      9291      9300      9303      9306      9316      9318      9322      9324      9338
              9539      9573      9607      9642      9673      9702      9714      9727      9741      9755      9844      9900      9909      9912
              9915      9918      9924      9951      9965      9968      9985      9988      9990     10001     10005     10007     10011     10013
             10023     10027     10029     10033     10035     10045     10049     10051     10055     10057
TMSGCD         695       807      1344      1346      1348      1388      1706      1707      1708      3298      3663      3687      4087      4159
              6580      6605      6789      7919      8370      8688      9013      9513     10178
TMSGD          800       802       804       806      1390      1400      1412      1417      1425      3300      3315      3320      3325      3329
              3339      3344      3349      3354      3362      3367      3372      3379      3385      3391      3399      3407      3528      3530
              3552      3554      3578      4109      4129      4333      6261      6319      6326      6366
TTALTM        1037      1870      3754      3856      3872      3885      3902      3930      4121      4161      4883      4907      5079      5247
              5369      5529      5804      6220      6313      6357      6414      6442      6672      6748      7367
TTIDEC        1737
TTSIXB         399      2878      4638
```

```
                                TITLE   DFPTA2
                                SUBTTL  EBUS Module Tests

                                SALL

                                SEARCH  DFPTA,MONSYM

                        ; INTERN's

                                ENTRY   TSTE1,TSTE2,TSTE3,TSTE4,TSTE5,TSTE6,TSTE7,TSTE10
                                ENTRY   TSTE11,TSTE12,TSTE13,TSTE14,TSTE15,TSTE16,TSTE17,TSTE20
                                ENTRY   TSTE21,TSTE22,TSTE23,TSTE24,TSTE25,TSTE26,TSTE27,TSTE30
                                ENTRY   TSTE31,TSTE32,TSTE33,TSTE34,TSTE35,TSTE36,TSTE37,TSTE40
                                ENTRY   TSTE41,TSTE42,TSTE43,TSTE44,TSTE45,TSTE46,TSTE47,TSTE50
                                ENTRY   TSTE51,TSTE52,TSTE53,TSTE54,TSTE55,TSTE56,TSTE57,TSTE60
                                ENTRY   TSTE61,TSTE62,TSTE63,TSTE64,TSTE65,TSTE66,TSTE67,TSTE70
                                ENTRY   TSTE71,TSTE72,TSTE73,TSTE74,TSTE75,TSTE76

                        ; EXTERN's

                                EXTERN  TSTA1,TSTA2,TSTA3,TSTA4,TSTA5,TSTA6,TSTA7
                                EXTERN  TSTU6,TSTU7,TSTU10,TSTU11,TSTU12,TSTU13,TSTU41

                        ; EXTERN's located in DFPTA1.MAC (Mainline code..)

                                EXTERN  TSTFLG,TSTSUB,TSTPC,UDEBUG,BUFF
                                EXTERN  TLOAD,CSRPNT,CSRENG,TRACE,ODELAY
                                EXTERN  IEXEC,IIPNT,ICSR,IADDR,BEXEC,BBPNT,CEBUF,AEBUF,IFLAG,IIOPF

                        ; EXTERN's located in DFPTAI.MAC (Port Device Handling Routines module)

                                EXTERN  LDEBUF,LDRAR,LDCSR,RDEBUF,RDCSR,ERESET,IPACLR
                                EXTERN  IPASRT,SNEXT,SDATA,.DATAI,.DATAO
                                EXTERN  INITPI,.PION,SETVEC,.CONI,.CONO,INTNUM,INTTYP
                                EXTERN  PFAPR,PFAIL,PF500,PF501,PF502,PISYOF,PISYON

                        ; EXTERN's located in DFPTAU.MAC (Utility Routines module)

                                EXTERN  SCOSW,BUFGEN

                        ;#*********************************************************************
                        ; Z2 - Address for use in DDT
                        ;#*********************************************************************

000000'                 Z2:                                     ; address of 00000'
```

```

                                       ;#*********************************************************************
                                       ;* TEST 1 - Is there an port out there?
                                       ;
                                       ; Description:  Just do a CONI and see if there is any device out
                                       ;               there at all.
                                       ;
                                       ; Procedure:    EBUS Reset
                                       ;               Read CSR
                                       ;               Data?  non-zero - ok
                                       ;                      zero - failed
                                       ;
                                       ; Failure:      If the CONI CSR returns zero, it is likely that
                                       ;               the CONI is failing and timeout has occurred.
                                       ;#*********************************************************************

                                       ; Test data

000000' 254 00 0 00 000005'            TSTE1:  JRST    TG1                     ; go start test
000001' 400000  000001                         EBUS!ZEBUS!1                    ; test mask
000002' 000000  010374'                        0,,[ASCIZ /CONI Works?/]        ; ucode,,description
000003' 010377' 010401'                        [EXP E1,MLAST!E22],,[EXP E9,MLAST!E5]
000004' 777777  777777                         -1                              ; failure test table

                                       ; Start test

000005' 201 00 0 00 000000'            TG1:    MOVEI   Z2                      ; get address of module start
000006' 260 17 0 00 000000*                    GO      TRACE                   ; handle trace output

000007' 400 15 0 00 000000             TA1:    SETZ    ERFLG,                  ; clear error flag
000010' 260 17 0 00 000000*                    GO      ERESET                  ; do an EBUS Reset
000011' 260 17 0 00 000000*                    GO      RDCSR                   ; read CSR
000012' 474 15 0 00 000000                     SETO    ERFLG,                  ; failed - set error flag
000013' 336 00 0 00 000001                     SKIPN   1                       ; result zero?
000014' 474 15 0 00 000000                     SETO    ERFLG,                  ; yes - set error flag
000015' 027 00 0 00 000021'                    SCOPER  MA1                     ; print error message
000016' 254 00 0 00 000007'                    JRST    TA1                     ; loop on error
000017' 254 00 0 00 000020'                    JRST    TX1                     ; altmode exit

                                       ; End of test

000020' 263 17 0 00 000000             TX1:    RTN                             ; return

                                       ; Error messages

000021' 140000  010403'                MA1:    MSG!TXNOT![ASCIZ /Did EBUS Reset, then CONI CSR - Should be non-zero./]
000022' 140000  010416'                        MSG!TXNOT![ASCIZ /CSR - Actual:  000000,,000000/]
000023' 140000  010424'                        MSG!TXNOT![ASCIZ /     Correct:  400003,,000000/]
000024' 130000  010432'                        LAST!MSG!TXYES![ASCIZ /CSR zero after EBUS Reset/]
```

```

                                       ;#*******************************************************************************
                                       ;* Test 2 - RH20/Port Interaction
                                       ;
                    .                  ; Description:  Check for RH20/Port Interaction problems by  doing
                    .                  ;               a reset, then  doing something to an RH20 and then
                    .                  ;               seeing if something can be done to the port.
                                       ;
                                       ; Procedure:    EBUS Reset
                                       ;               Read CSR and save results
                                       ;               CONI RH0
                                       ;               Read CSR
                                       ;               CONI data should be the same as the first
                                       ;                 CONI - if so - ok,
                                       ;                        if not - error
                                       ;
                                       ;               Repeat the EBUS Reset/CONI/CONI sequence
                                       ;                 for each RH1-7 except the port(s).
                                       ;
                                       ; Failure:      The CONI RHx may be decoded and executed partially
                                       ;               by the port.  The second CONI then returns incorrect
                                       ;               data because it is completing a partially done CONI.
                                       ;#*******************************************************************************

                                       ; Test data

000025' 254 00 0 00 000035'            TSTE2:  JRST    TG2                             ; go start test
000026' 400000  000002                         EBUS!ZEBUS!2                            ; test mask
000027' 000000  010440'                        0,,[ASCIZ ^RH20/Port Interaction^]
000030' 010377' 010401'                        [EXP E1,MLAST!E22],,[EXP E9,MLAST!E5]
000031' 000000  000123'                        TSTE3                                   ; failure test table
000032' 000000  000214'                        TSTE4                                   ;   ...
000033' 000000  000307'                        TSTE5
000034' 777777  777777                         -1

                                       ; Start test

000035' 201 00 0 00 000000'            TG2:    MOVEI   Z2                              ; get address of module start
000036' 260 17 0 00 000006*                    GO      TRACE                           ; handle trace output
000037' 260 17 0 00 000010*                    GO      ERESET                          ; do an EBUS Reset
000040' 202 16 0 00 013666'                    MOVEM   MBCN,SAVMBC#                    ; save the MBC number
000041' 200 16 0 00 010445'                    MOVE    MBCN,[54000,,0]                 ; initialize RH number
```

```

                                        ; Segment 0-7 of test

000042' 350 00 0 00 000000*             TA2:    AOS     TSTSUB                  ; indicate next subtest
000043' 400 15 0 00 000000              TA2L:   SETZ    ERFLG,                  ; clear error flag
000044' 316 16 0 00 013666'                     CAMN    MBCN,SAVMBC             ; is this the port under test?
000045' 254 00 0 00 000070'                     JRST    TA2X                    ; yes - do next RH

000046' 250 16 0 00 013666'                     EXCH    MBCN,SAVMBC             ; set up to read port
000047' 260 17 0 00 000011*                     GO      RDCSR                   ; read CSR
000050' 255 00 0 00 000000                      JFCL                            ; ignore error
000051' 250 16 0 00 013666'                     EXCH    MBCN,SAVMBC             ; restore MBC number
000052' 202 01 0 00 010362'                     MOVEM   1,SAVDA1                ; save result
000053' 260 17 0 00 000000*                     GO      .CONI                   ; do a CONI of RH x
000054' 202 01 0 00 010363'                     MOVEM   1,SAVDA2                ; save result
000055' 250 16 0 00 013666'                     EXCH    MBCN,SAVMBC             ; set up to read port
000056' 260 17 0 00 000047*                     GO      RDCSR                   ; read CSR
000057' 255 00 0 00 000000                      JFCL                            ; ignore error
000060' 250 16 0 00 013666'                     EXCH    MBCN,SAVMBC             ; restore MBC number
000061' 202 01 0 00 010364'                     MOVEM   1,SAVDA3                ; save result
000062' 312 01 0 00 010362'                     CAME    1,SAVDA1                ; data same as before?
000063' 474 15 0 00 000000                      SETO    ERFLG,                  ; no - set error flag
000064' 027 00 0 00 000075'                     SCOPER  MA2                     ; print error message
000065' 254 00 0 00 000043'                     JRST    TA2L                    ; loop on error
000066' 254 00 0 00 000073'                     JRST    TX2                     ; altmode exit
000067' 326 15 0 00 000073'                     JUMPN   ERFLG,TX2               ; error? yes - exit

000070' 270 16 0 00 010446'             TA2X:   ADD     MBCN,[400,,0]           ; point to next RH
000071' 317 16 0 00 010447'                     CAMG    MBCN,[57400,,0]         ; reach RH6 yet?
000072' 254 00 0 00 000042'                     JRST    TA2                     ; no - loop till done

                                        ; End of test

000073' 200 16 0 00 013666'             TX2:    MOVE    MBCN,SAVMBC             ; restore MBC number
000074' 263 17 0 00 000000                      RTN                             ; return
```

```

                                        ; Error messages

000075' 140000  010450'                 MA2:    MSG!TXNOT![ASCIZ /Did EBUS Reset, then CONI CSR, then CONI RH/]
000076' 240000  000102'                         CALL!TXNOT!MA2PNA
000077' 140000  010461'                         MSG!TXNOT![ASCIZ /Both CONI CSR's should be the same./]
000100' 240000  000106'                         CALL!TXNOT!MA2PNB
000101' 230000  010224'                         LAST!CALL!TXYES!MPNT1           ; Port/RH Interaction Message

000102' 135 00 0 00 010471'             MA2PNA: LDB     [POINT 3,MBCN,9]        ; get basic 3 bits
000103' 037 16 0 00 000003                      PNTOCS
000104' 037 00 0 00 010472'                     TMSG    <, then CONI CSR.>
000105' 263 17 0 00 000000                      RTN

000106' 037 00 0 00 010476'             MA2PNB: TMSGC   <CONI RH>
000107' 135 00 0 00 010471'                     LDB     [POINT 3,MBCN,9]        ; get basic 3 bits
000110' 037 16 0 00 000003                      PNTOCS
000111' 037 00 0 00 010500'                     TMSG    <:  >
000112' 200 00 0 00 010363'                     MOVE    SAVDA2
000113' 037 13 0 00 000000                      PNTHW
000114' 037 00 0 00 010501'                     TMSGC   <CSR (Original):  >
000115' 200 00 0 00 010362'                     MOVE    SAVDA1
000116' 037 13 0 00 000000                      PNTHW
000117' 037 00 0 00 010505'                     TMSGC   <  (After CONI):  >
000120' 200 00 0 00 010364'                     MOVE    SAVDA3
000121' 037 13 0 00 000000                      PNTHW
000122' 263 17 0 00 000000                      RTN                             ; return
```

```

                                ;#*******************************************************************
                                ;* Test 3 - RH20/Port Interaction
                                ;
                                ; Description:  Check for RH20/Port Interaction problems by doing
                                ;               a reset, then doing something to  the  port, then
                                ;               seeing if something can be done to an RH20.
                                ;
                                ; Procedure:    EBUS Reset
                                ;               CONI RHO and save results
                                ;               Read CSR
                                ;               CONI RHO
                                ;               CONI data should be the same as the first
                                ;                 CONI - if so - ok
                                ;                        if not - error
                                ;
                                ;               Repeat this sequence for each RH1-7 except the port(s).
                                ;
                                ; Failure:      The CONI CSR may leave the EBUS Control logic in
                                ;               a funny state that causes successive CONI's to
                                ;               other devices to fail.
                                ;#*******************************************************************

                                ; Test data

000123' 254 00 0 00 000132'     TSTE3:  JRST    TG3                     ; go start test
000124' 400000  000003                  EBUS!ZEBUS!3                    ; test mask
000125' 000000  010440'                 0,,[ASCIZ ^RH20/Port Interaction^]
000126' 010377' 010401'                 [EXP E1,MLAST!E22],,[EXP E9,MLAST!E5]
000127' 000000  000214'                 TSTE4                           ; failure test table
000130' 000000  000307'                 TSTE5                           ;   ...
000131' 777777  777777                  -1

                                ; Start test

000132' 332 00 0 00 030037      TG3:    SKIPE   USER                    ; user mode?
                                        GO      [MOVE   TSTFLG          ; yes - ensure this test does not
                                                 TLO    (TUSER)         ;    get run (but only if not in
                                                 SKIPN  UDEBUG          ;    debug mode)
                                                 MOVEM  TSTFLG
000133' 260 17 0 00 010511'                      RTN]
000134' 201 00 0 00 000000'             MOVEI   Z2                      ; get address of module start
000135' 260 17 0 00 000036*             GO      TRACE                   ; handle trace output
000136' 260 17 0 00 000037*             GO      ERESET                  ; do an EBUS Reset
000137' 202 16 0 00 013666'             MOVEM   MBCN,SAVMBC             ; save the MBC number
000140' 200 16 0 00 010445'             MOVE    MBCN,[54000,,0]         ; initialize RH number

                                ; Segment 0-7 of test

000141' 350 00 0 00 000042*     TA3:    AOS     TSTSUB                  ; indicate next subtest
000142' 400 15 0 00 000000      TA3L:   SETZ    ERFLG,                  ; clear error flag
```

```
000143' 316 16 0 00 013666'             CAMN    MBCN,SAVMBC             ; is this the port?
000144' 254 00 0 00 000163'             JRST    TA3X                    ; yes - do next RH
000145' 260 17 0 00 000053*             GO      .CONI                   ; do a CONI of RH x
000146' 202 01 0 00 010362'             MOVEM   1,SAVDA1                ; save result
000147' 250 16 0 00 013666'             EXCH    MBCN,SAVMBC             ; set up to read port
000150' 260 17 0 00 000056*             GO      RDCSR                   ; read CSR
000151' 255 00 0 00 000000              JFCL                            ; ignore error
000152' 250 16 0 00 013666'             EXCH    MBCN,SAVMBC             ; restore MBC number
000153' 260 17 0 00 000145*             GO      .CONI                   ; do a CONI of RH x
000154' 202 01 0 00 010363'             MOVEM   1,SAVDA2                ; save result
000155' 312 01 0 00 010362'             CAME    1,SAVDA1                ; data same as before?
000156' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
000157' 027 00 0 00 000170'             SCOPER  MA3                     ; print error message
000160' 254 00 0 00 000142'             JRST    TA3L                    ; loop on error
000161' 254 00 0 00 000166'             JRST    TX3                     ; altmode exit
000162' 326 15 0 00 000166'             JUMPN   ERFLG,TX3               ; error? yes - exit

000163' 270 16 0 00 010446'     TA3X:   ADD     MBCN,[400,,0]           ; point to next RH
000164' 317 16 0 00 010447'             CAMG    MBCN,[57400,,0]         ; reach RH6 yet?
000165' 254 00 0 00 000141'             JRST    TA3                     ; no - loop till done

                                ; End of test

000166' 200 16 0 00 013666'     TX3:    MOVE    MBCN,SAVMBC             ; restore MBC number
000167' 263 17 0 00 000000              RTN                             ; return

                                ; Error messages

000170' 240000  000172'         MA3:    CALL!TXNOT!MA3PNT
000171' 230000  010224'                 LAST!CALL!TXYES!MPNT1           ; Port/RH Interaction Message

000172' 037 00 0 00 010516'     MA3PNT: TMSGC   <Did EBUS Reset, then CONI RH>
000173' 135 00 0 00 010471'             LDB     [POINT 3,MBCN,9]        ; get basic 3 bits
000174' 037 16 0 00 000003              PNTOCS
000175' 037 00 0 00 010525'             TMSG    <, then CONI CSR, then CONI RH>
000176' 135 00 0 00 010471'             LDB     [POINT 3,MBCN,9]        ; get basic 3 bits
000177' 037 16 0 00 000003              PNTOCS
000200' 037 00 0 00 010415'             TMSG    <.>
000201' 037 00 0 00 010533'             TMSGC   <Both CONI RH's should be the same.>

000202' 037 00 0 00 010476'             TMSGC   <CONI RH>
000203' 135 00 0 00 010471'             LDB     [POINT 3,MBCN,9]        ; get basic 3 bits
000204' 037 16 0 00 000003              PNTOCS
000205' 037 00 0 00 010502'             TMSG    < (Original):  >
000206' 200 00 0 00 010362'             MOVE    SAVDA1
000207' 037 13 0 00 000000              PNTHW
000210' 037 00 0 00 010543'             TMSGC   <       (After RdCSR):  >
000211' 200 00 0 00 010363'             MOVE    SAVDA2
000212' 037 13 0 00 000000              PNTHW
000213' 263 17 0 00 000000              RTN                             ; return
```

```
                                        ;#*********************************************************************
                                        ;* Test 4 - RH20/Port Interaction
                                        ;
                                        ; Description:  Check for RH20/Port Interaction problems by doing
                                        ;               a reset, then doing something to an RH20 and then
                                        ;               seeing if something can be done to the port.
                                        ;
                                        ; Procedure:    EBUS Reset
                                        ;               Read CSR and save results
                                        ;               CONO RH0
                                        ;               Read CSR
                                        ;               CSR data should be the same as the first
                                        ;                 CSR - if so - ok
                                        ;                       if not - error
                                        ;
                                        ;               Repeat this sequence for each RH1-7 except the port(s).
                                        ;
                                        ; Failure:      The CONO RH0 may be decoded and executed partially
                                        ;               by the port.  The CONI then returns incorrect data
                                        ;               because it is completing a partially done CONO.
                                        ;#*********************************************************************

                                        ; Test data

000214' 254 00 0 00 000222'             TSTE4:  JRST    TG4                     ; go start test
000215' 400000  000004                          EBUS!ZEBUS!4                    ; test mask
000216' 000000  010440'                         0,,[ASCIZ ^RH20/Port Interaction^]
000217' 010377' 010401'                         [EXP E1,MLAST!E22],,[EXP E9,MLAST!E5]
000220' 000000  000307'                         TSTE5                           ; failure test table
000221' 777777  777777                          -1                              ;   ...

                                        ; Start test

000222' 332 00 0 00 030037              TG4:    SKIPE   USER                    ; user mode?
                                                GO      [MOVE   TSTFLG          ; yes - ensure this test does not
                                                         TLO    (TUSER)         ;    get run (but only if not in
                                                         SKIPN  UDEBUG          ;    debug mode)
                                                         MOVEM  TSTFLG
000223' 260 17 0 00 010511'                              RTN]
000224' 201 00 0 00 000000'                     MOVEI   Z2                      ; get address of module start
000225' 260 17 0 00 000135*                     GO      TRACE                   ; handle trace output
000226' 202 16 0 00 013666'                     MOVEM   MBCN,SAVMBC             ; save the MBC number
000227' 200 16 0 00 010445'                     MOVE    MBCN,[54000,,0]         ; initialize RH number

                                        ; Segment 0-7 of test

000230' 350 00 0 00 000141*             TA4:    AOS     TSTSUB                  ; indicate next subtest
000231' 400 15 0 00 000000              TA4L:   SETZ    ERFLG,                  ; clear error flag
000232' 316 16 0 00 013666'                     CAMN    MBCN,SAVMBC             ; is this the port?
000233' 254 00 0 00 000257'                     JRST    TA4X                    ; yes - do next RH
```

```
000234' 260 17 0 00 000136*          GO      ERESET                  ; do an EBUS Reset
000235' 250 16 0 00 013666'          EXCH    MBCN,SAVMBC             ; set up to read port
000236' 260 17 0 00 000150*          GO      RDCSR                   ; read CSR
000237' 255 00 0 00 000000           JFCL                            ; ignore error
000240' 202 01 0 00 010362'          MOVEM   1,SAVDA1                ; save result
000241' 250 16 0 00 013666'          EXCH    MBCN,SAVMBC             ; restore MBC number
000242' 201 01 0 00 002000           MOVEI   1,2000                  ; set some RH bits
000243' 260 17 0 00 000000*          GO      .CONO                   ; do a CONO of RH x
000244' 250 16 0 00 013666'          EXCH    MBCN,SAVMBC             ; set up to read port
000245' 260 17 0 00 000236*          GO      RDCSR                   ; read CSR
000246' 255 00 0 00 000000           JFCL                            ; ignore error
000247' 202 01 0 00 010363'          MOVEM   1,SAVDA2                ; save result
000250' 250 16 0 00 013666'          EXCH    MBCN,SAVMBC             ; restore MBC number
000251' 312 01 0 00 010362'          CAME    1,SAVDA1                ; data same as before?
000252' 474 15 0 00 000000           SETO    ERFLG,                  ; no - set error flag
000253' 027 00 0 00 000264'          SCOPER  MA4                     ; print error message
000254' 254 00 0 00 000231'          JRST    TA4L                    ; loop on error
000255' 254 00 0 00 000262'          JRST    TX4                     ; altmode exit
000256' 326 15 0 00 000262'          JUMPN   ERFLG,TX4               ; error? yes - exit

000257' 270 16 0 00 010446'  TA4X:   ADD     MBCN,[400,,0]           ; point to next RH
000260' 317 16 0 00 010447'          CAMG    MBCN,[57400,,0]         ; reach RH6 yet?
000261' 254 00 0 00 000230'          JRST    TA4                     ; no - loop till done

                             ; End of test

000262' 200 16 0 00 013666'  TX4:    MOVE    MBCN,SAVMBC             ; restore MBC number
000263' 263 17 0 00 000000           RTN                             ; return

                             ; Error messages

000264' 140000  010550'      MA4:    MSG!TXNOT![ASCIZ /Did EBUS Reset, then CONI CSR, then CONO RH/]
000265' 240000  000271'              CALL!TXNOT!MA4PNA
000266' 140000  010561'              MSG!TXNOT![ASCIZ /  CONI CSR.  Both CONI CSR's should be the same./]
000267' 240000  000275'              CALL!TXNOT!MA4PNB
000270' 230000  010224'              LAST!CALL!TXYES!MPNT1           ; Port/RH Interaction Message

000271' 135 00 0 00 010471'  MA4PNA: LDB     [POINT 3,MBCN,9]
000272' 037 16 0 00 000003           PNTOCS
000273' 037 00 0 00 010573'          TMSG    <, then>
000274' 263 17 0 00 000000           RTN

000275' 037 00 0 00 010575'  MA4PNB: TMSGC   <CONO on RH>
000276' 135 00 0 00 010471'          LDB     [POINT 3,MBCN,9]        ; get basic 3 bits
000277' 037 01 0 00 000000           PNT1
000300' 037 00 0 00 010501'          TMSGC   <CSR (Original):  >
000301' 200 00 0 00 010362'          MOVE    SAVDA1
000302' 037 13 0 00 000000           PNTHW
000303' 037 00 0 00 010600'          TMSGC   <  (After CONO):  >
000304' 200 00 0 00 010363'          MOVE    SAVDA2
000305' 037 13 0 00 000000           PNTHW
000306' 263 17 0 00 000000           RTN                             ; return
```

```

                                        ;#*****************************************************************
                                        ;* Test 5 - RH20/Port Interaction
                                        ;
                                        ; Description:  Check for RH20/Port Interaction problems by doing
                                        ;               a reset, then doing something to  the  port, then
                                        ;               seeing if something can be done to an RH20.
                                        ;
                                        ; Procedure:    EBUS Reset
                                        ;               CONI RHO and save results
                                        ;               Write CSR (Set some bits)
                                        ;               CONI RHO
                                        ;               CONI data should be the same as the first
                                        ;                 CONI - if so - ok
                                        ;                        if not - error
                                        ;
                                        ;               Repeat this sequence for each RH1-7 except the port(s).
                                        ;
                                        ; Failure:      The CONO CSR may leave the EBUS Control logic in
                                        ;               a funny state that causes successive CONI's to
                                        ;               other devices to fail.
                                        ;#*****************************************************************

                                        ; Test data

000307' 254 00 0 00 000314'             TSTE5:  JRST    TG5                     ; go start test
000310' 400000  000005                          EBUS!ZEBUS!5                    ; test mask
000311' 000000  010440'                         0,,[ASCIZ ^RH20/Port Interaction^]
000312' 010377' 010401'                         [EXP E1,MLAST!E22],,[EXP E9,MLAST!E5]
000313' 777777  777777                          -1                              ; failure test table

                                        ; Start test

000314' 332 00 0 00 030037              TG5:    SKIPE   USER                    ; user mode?
                                                GO      [MOVE   TSTFLG          ; yes - ensure this test does not
                                                         TLO    (TUSER)         ;    get run (but only if not in
                                                         SKIPN  UDEBUG          ;    debug mode)
                                                         MOVEM  TSTFLG
000315' 260 17 0 00 010511'                              RTN]
000316' 201 00 0 00 000000'                     MOVEI   Z2                      ; get address of module start
000317' 260 17 0 00 000225*                     GO      TRACE                   ; handle trace output
000320' 260 17 0 00 000234*                     GO      ERESET                  ; do an EBUS Reset
000321' 202 16 0 00 013666'                     MOVEM   MBCN,SAVMBC             ; save the MBC number
000322' 200 16 0 00 010445'                     MOVE    MBCN,[54000,,0]         ; initialize RH number

                                        ; Segment 0-7 of test

000323' 350 00 0 00 000230*             TA5:    AOS     TSTSUB                  ; indicate next subtest
000324' 400 15 0 00 000000              TA5L:   SETZ    ERFLG,                  ; clear error flag
000325' 316 16 0 00 010447'                     CAMN    MBCN,[57400,,0]         ; is this the port?
000326' 254 00 0 00 000345'                     JRST    TA5X                    ; yes - do next RH

```

```
000327' 260 17 0 00 000153*              GO      .CONI                   ; do a CONI of RH x
000330' 202 01 0 00 010362'              MOVEM   1,SAVDA1                ; save result
000331' 250 16 0 00 013666'              EXCH    MBCN,SAVMBC             ; set up to read port
000332' 201 01 0 00 263007               MOVEI   1,263007                ; write some bits to
000333' 260 17 0 00 000000*              GO      LDCSR                   ;   CSR register
000334' 250 16 0 00 013666'              EXCH    MBCN,SAVMBC             ; restore MBC number
000335' 260 17 0 00 000327*              GO      .CONI                   ; do a CONI of RH x
000336' 202 01 0 00 010363'              MOVEM   1,SAVDA2                ; save result
000337' 312 01 0 00 010362'              CAME    1,SAVDA1                ; data same as before?
000340' 474 15 0 00 000000               SETO    ERFLG,                  ; no - set error flag
000341' 027 00 0 00 000352'              SCOPER  MA5                     ; print error message
000342' 254 00 0 00 000324'              JRST    TA5L                    ; loop on error
000343' 254 00 0 00 000350'              JRST    TX5                     ; altmode exit
000344' 326 15 0 00 000350'              JUMPN   ERFLG,TX5               ; error? yes - exit

000345' 270 16 0 00 010446'      TA5X:   ADD     MBCN,[400,,0]           ; point to next RH
000346' 317 16 0 00 010447'              CAMG    MBCN,[57400,,0]         ; reach RH6 yet?
000347' 254 00 0 00 000323'              JRST    TA5                     ; no - loop till done

                                 ; End of test

000350' 200 16 0 00 013666'      TX5:    MOVE    MBCN,SAVMBC             ; restore MBC number
000351' 263 17 0 00 000000               RTN                             ; return

                                 ; Error messages

000352' 240000  000354'          MA5:    CALL!TXNOT!MA5PNT
000353' 230000  010224'                  LAST!CALL!TXYES!MPNT1           ; Port/RH Interaction Message

000354' 037 00 0 00 010516'      MA5PNT: TMSGC   <Did EBUS Reset, then CONI RH>
000355' 135 00 0 00 010471'              LDB     [POINT 3,MBCN,9]
000356' 037 16 0 00 000003               PNTOCS
000357' 037 00 0 00 010604'              TMSG    <, then CONO CSR, then>
000360' 037 00 0 00 010611'              TMSGC   <  CONI RH>
000361' 135 00 0 00 010471'              LDB     [POINT 3,MBCN,9]
000362' 037 16 0 00 000003               PNTOCS
000363' 037 00 0 00 010614'              TMSG    <.  Both CONI RH's should be the same.>

000364' 037 00 0 00 010476'              TMSGC   <CONI RH>
000365' 135 00 0 00 010471'              LDB     [POINT 3,MBCN,9]        ; get basic 3 bits
000366' 037 16 0 00 000003               PNTOCS
000367' 037 00 0 00 010502'              TMSG    < (Original):  >
000370' 200 00 0 00 010362'              MOVE    SAVDA1
000371' 037 13 0 00 000000               PNTHW
000372' 037 00 0 00 010624'              TMSGC   <        (After LDCSR):  >
000373' 200 00 0 00 010363'              MOVE    SAVDA2
000374' 037 13 0 00 000000               PNTHW
000375' 263 17 0 00 000000               RTN                             ; return
```

0465

```

                                          ;#*****************************************************************
                                          ;* Test 6 - CSR Cleared After Reset
                                          ;
                                          ; Description:  Ensure that an EBUS Reset signal propagates  from
                                          ;               the CBUS to CBus Module and to reset logic on the
                                          ;               EBus Module correctly.
                                          ;
                                          ; Procedure:    EBUS Reset
                                          ;               Read CSR
                                          ;               Correct bits set?  yes - ok
                                          ;                                  no - error
                                          ;
                                          ; Failure:      Either the reset failed to clear the CSR Register
                                          ;               or it is not possible to read the CSR Register
                                          ;               correctly.
                                          ;#*****************************************************************

                                          ; Test data

000376' 254 00 0 00 000410'               TSTE6:  JRST    TG6                     ; go start test
000377' 400000  000006                            EBUS!ZEBUS!6                    ; test mask
000400' 000000  010631'                           0,,[ASCIZ ^CSR Cleared After Reset^]
000401' 010636' 000000                            [EXP E6,E1,E21,E22,E9,E5,E11,E12,E13,E14,E15,E16,MLAST!E17],,0
000402' 000000  000432'                           TSTE7                           ; failure test table
000403' 000000  000616'                           TSTE10                          ;   ...
000404' 000000  000704'                           TSTE11
000405' 000000  000735'                           TSTE12
000406' 000000  001106'                           TSTE14
000407' 777777  777777                            -1

                                          ; Start test

000410' 332 00 0 00 030037                TG6:    SKIPE   USER                    ; user mode?
                                                  GO      [MOVE   TSTFLG          ; yes - ensure this test does not
                                                           TLO    (TUSER)         ;    get run (but only if not in
                                                           SKIPN  UDEBUG          ;    debug mode)
                                                           MOVEM  TSTFLG
000411' 260 17 0 00 010511'                                RTN]
000412' 201 00 0 00 000000'                       MOVEI   Z2                      ; get address of module start
000413' 260 17 0 00 000317*                       GO      TRACE                   ; handle trace output
000414' 200 00 0 00 010653'                       MOVE    [400003,,0]             ; set up correct data
000415' 202 00 0 00 010367'                       MOVEM   SAVCS2                  ; save it
```

```

000416' 400 15 0 00 000000      TA6:    SETZ    ERFLG,                  ; clear error flag
000417' 260 17 0 00 000320*             GO      ERESET                  ; do an EBUS reset
000420' 260 17 0 00 000335*             GO      .CONI                   ; read CSR register
000421' 202 01 0 00 010370'             MOVEM   1,SAVCS3                ; save data
000422' 312 01 0 00 010367'             CAME    1,SAVCS2                ; correct bits set?
000423' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
000424' 027 00 0 00 000430'             SCOPER  MA6                     ; print error message
000425' 254 00 0 00 000416'             JRST    TA6                     ; loop on error
000426' 254 00 0 00 000427'             JRST    TX6                     ; altmode exit

                                ; End of test

000427' 263 17 0 00 000000      TX6:    RTN                             ; return

                                ; Error messages

000430' 140000  010654'         MA6:    MSG!TXNOT![ASCIZ /Do EBUS Reset, then read CSR.  CSR incorrect./]
000431' 270000  010254'                 LAST!CALL!TXALL!MPNT5           ; print CSR contents
```

```

                                ;#***********************************************************************
                                ;* Test 7 - Read/Write CSR Bits
                                ;
                                ; Description:  Ensure that CSR bits can  be  read  and  written.
                                ;               Each bit  including  the  read/write only bits is
                                ;               written.  The  CSR  Register  is  then  read  and
                                ;               verified, then the  bit  is  cleared  and the CSR
                                ;               Register read and verified again.
                                ;
                                ; Procedure:    EBUS Reset
                                ;               Select bit 18
                                ;               Write CSR (set a bit)
                                ;               Read CSR (read the bit)
                                ;               Correct result read?  if not - error
                                ;               Write CSR (clear the bit)
                                ;               Read CSR (read the bit)
                                ;               Correct result read?  if not - error
                                ;
                                ;               Repeat test for each bit 18-35
                                ;
                                ; Failure:      ---
                                ;#***********************************************************************

                                ; Test data

000432' 254 00 0 00 000440'     TSTE7:  JRST    TG7                             ; go start test
000433' 400000  000007                  EBUS!ZEBUS!7                            ; test mask
000434' 000000  010666'                 0,,[ASCIZ ^Read/Write CSR Bits^]
000435' 010672' 000000                  [EXP E6,E1,E21,E22,E9,E11,E12,E13,E14,E15,E16,MLAST!E17],,0
000436' 000000  000616'                 TSTE10                                  ; failure test table
000437' 777777  777777                  -1                                      ;   ...

                                ; Start test

000440' 201 00 0 00 000000'     TG7:    MOVEI   Z2                              ; get address of module start
000441' 260 17 0 00 000413*             GO      TRACE                           ; handle trace output
000442' 400 06 0 00 000000              SETZ    6,                              ; clear bit table pointer

                                ; First, set the bit

000443' 400 15 0 00 000000      TA7:    SETZ    ERFLG,                          ; clear error flag
000444' 200 00 0 00 000006              MOVE    6                               ; set up subtest number as
000445' 231 00 0 00 000003              IDIVI   3                               ;   pattern number * 2 + 1
000446' 242 00 0 00 000001              LSH     1                               ;   and save
000447' 271 00 0 00 000001              ADDI    1
000450' 202 00 0 00 000323*             MOVEM   TSTSUB
000451' 336 01 0 06 000530'             SKIPN   1,TA7PAT(6)                     ; get bit to write
000452' 254 00 0 00 000511'             JRST    TX7                             ; zero - end of test

```

```
000453' 260 17 0 00 000417*         GO      ERESET                  ; do an EBUS reset
000454' 202 01 0 00 010366'         MOVEM   1,SAVCS1                ; save data
000455' 260 17 0 00 000333*         GO      LDCSR                   ; write data
000456' 260 17 0 00 000245*         GO      RDCSR                   ; read CSR
000457' 255 00 0 00 000000          JFCL                            ; ignore error
000460' 202 01 0 00 010370'         MOVEM   1,SAVCS3                ; save data
000461' 200 00 0 06 000531'         MOVE    TA7PAT+1(6)             ; get correct data
000462' 202 00 0 00 010367'         MOVEM   SAVCS2                  ; save data
000463' 312 00 0 00 000001          CAME    1                       ; actual same as correct?
000464' 474 15 0 00 000000          SETO    ERFLG,                  ; no - set error flag
000465' 027 00 0 00 000512'         SCOPER  MA7                     ; print error message
000466' 254 00 0 00 000443'         JRST    TA7                     ; loop on error
000467' 254 00 0 00 000511'         JRST    TX7                     ; altmode exit
000470' 326 15 0 00 000511'         JUMPN   ERFLG,TX7               ; error? yes - exit

                            ; Then, clear the bit

000471' 350 00 0 00 000450*         AOS     TSTSUB                  ; point to next subtest
000472' 400 01 0 00 000000          SETZ    1,                      ; zero
000473' 260 17 0 00 000455*         GO      LDCSR                   ; clear the bit
000474' 260 17 0 00 000456*         GO      RDCSR                   ; read CSR
000475' 255 00 0 00 000000          JFCL                            ; ignore error
000476' 202 01 0 00 010370'         MOVEM   1,SAVCS3                ; save data
000477' 200 00 0 06 000532'         MOVE    TA7PAT+2(6)             ; get correct data
000500' 202 00 0 00 010367'         MOVEM   SAVCS2                  ; save data
000501' 312 00 0 00 000001          CAME    1                       ; actual same as correct?
000502' 474 15 0 00 000000          SETO    ERFLG,                  ; no - set error flag
000503' 027 00 0 00 000521'         SCOPER  MB7                     ; print error message
000504' 254 00 0 00 000443'         JRST    TA7                     ; loop on error
000505' 254 00 0 00 000511'         JRST    TX7                     ; altmode exit
000506' 326 15 0 00 000511'         JUMPN   ERFLG,TX7               ; error? yes - exit

                            ; Increment bit table pointer

000507' 271 06 0 00 000003          ADDI    6,3                     ; point to next entry
000510' 254 00 0 00 000443'         JRST    TA7                     ; repeat this segment

                            ; End of test

000511' 263 17 0 00 000000  TX7:   RTN                             ; return
```

```

                                    ; Error messages

000512' 240000  000514'             MA7:    CALL!TXNOT!MA7PNT              ; print data written
000513' 270000  010254'                     LAST!CALL!TXALL!MPNT5          ; print CSR correct/actual

000514' 037 00 0 00 010706'         MA7PNT: TMSGC   <Write CSR '>
000515' 200 00 0 00 010366'                 MOVE    SAVCS1
000516' 037 06 0 00 000000                  PNT6
000517' 037 00 0 00 010711'                 TMSG    <', then read CSR, verify result.>
000520' 263 17 0 00 000000                  RTN

000521' 240000  000523'             MB7:    CALL!TXNOT!MB7PNT              ; print data written
000522' 270000  010254'                     LAST!CALL!TXALL!MPNT5          ; print CSR correct/actual

000523' 037 00 0 00 010706'         MB7PNT: TMSGC   <Write CSR '>
000524' 200 00 0 00 010366'                 MOVE    SAVCS1
000525' 037 06 0 00 000000                  PNT6
000526' 037 00 0 00 010720'                 TMSG    <', then clear CSR, verify result.>
000527' 263 17 0 00 000000                  RTN

000530' 000000  400000              TA7PAT: EXP     400000,400003000000,500003000000        ; bit 18
000531' 400003  000000
000532' 500003  000000
000533' 000000  200000                      EXP     200000,500003200000,500003000000        ; bit 19
000534' 500003  200000
000535' 500003  000000
000536' 000000  100000                      EXP     100000,500003104000,500003004000        ; bit 20
000537' 500003  104000
000540' 500003  004000
000541' 000000  040000                      EXP     040000,500003040000,500003000000        ; bit 21
000542' 500003  040000
000543' 500003  000000
000544' 000000  020000                      EXP     020000,500003020000,500003000000        ; bit 22
000545' 500003  020000
000546' 500003  000000
000547' 000000  010000                      EXP     010000,500003010000,500003000000        ; bit 23
000550' 500003  010000
000551' 500003  000000
000552' 000000  004000                      EXP     004000,500003000000,500003000000        ; bit 24
000553' 500003  000000
000554' 500003  000000
000555' 000000  002000                      EXP     002000,500003000000,500003000000        ; bit 25
000556' 500003  000000
000557' 500003  000000
000560' 000000  001000                      EXP     001000,500003000000,500003000000        ; bit 26
000561' 500003  000000
000562' 500003  000000
000563' 000000  000400                      EXP     000400,500003000400,500003000400        ; bit 27
000564' 500003  000400
000565' 500003  000400
000566' 000000  000200                      EXP     000200,500003000000,500003000000        ; bit 28
000567' 500003  000000
000570' 500003  000000
000571' 000000  000100                      EXP     000100,500003000000,500003000000        ; bit 29
```

```
   711  000572' 500003  000000
   712  000573' 500003  000000
   713  000574' 000000  000040                 EXP     000040,500003000040,500003000040        ; bit 30
   714  000575' 500003  000040
   715  000576' 500003  000040
   716  000577' 000000  000020                 EXP     000020,500003000020,500003000020        ; bit 31
   717  000600' 500003  000020
   718  000601' 500003  000020
   719  000602' 000000  000004                 EXP     000004,500003000004,500003000000        ; bit 32
   720  000603' 500003  000004
   721  000604' 500003  000000
   722  000605' 000000  000002                 EXP     000002,500003000002,500003000000        ; bit 33
   723  000606' 500003  000002
   724  000607' 500003  000000
   725  000610' 000000  000001                 EXP     000001,500003000001,500003000000        ; bit 34
   726  000611' 500003  000001
   727  000612' 500003  000000
   728  000613' 000000  000000                 EXP     000000,500003000000,500003000000        ; bit 35
   729  000614' 500003  000000
   730  000615' 500003  000000
```

```

                                    ;#*********************************************************************
                                    ;* TEST 10 - CSR Bits Interaction
                                    ;
                                    ; Description:  Check  for  interaction  between  bits of the CSR
                                    ;               Register. Write them all either ones or zeros and
                                    ;               verify the results. If any of these bits are tied
                                    ;               to other CSR bits, the interaction  will be seen.
                                    ;               Also, since  the  18  bits of data is put on each
                                    ;               halfword of the  EBUS,  interaction  between  the
                                    ;               left  and right halves of the CSR register may be
                                    ;               detected.
                                    ;
                                    ; Procedure:    EBUS Reset
                                    ;               Write CSR with all 1's (except for 'Port Clear'
                                    ;                 and 'MPRun' bits)
                                    ;               Read CSR and compare against what bits
                                    ;                 should be set - if same - ok
                                    ;                                 if not - error
                                    ;               Write CSR with all 0's
                                    ;               Read CSR and compare against what bits
                                    ;                 should be set - if same - ok
                                    ;                                 if not - error
                                    ;
                                    ; Failure:      Some interaction between the CSR bits occurred,
                                    ;               which probably was also seen by prior tests.
                                    ;#*********************************************************************

                                    ; Test data

000616' 254 00 0 00 000627'         TSTE10: JRST    TG10                            ; go start test
000617' 400000  000010                      EBUS!ZEBUS!10                           ; test mask
000620' 000000  010727'                     0,,[ASCIZ ^CSR Bits Interaction^]
000621' 010672' 000000                      [EXP E6,E1,E21,E22,E9,E11,E12,E13,E14,E15,E16,MLAST!E17],,0
000622' 000000  000704'                     TSTE11                                  ; failure test table
000623' 000000  000735'                     TSTE12                                  ;   ...
000624' 000000  001106'                     TSTE14
000625' 000000  001162'                     TSTE15
000626' 777777  777777                      -1

                                    ; Start test

000627' 201 00 0 00 000000'         TG10:   MOVEI   Z2                              ; get address of module start
000630' 260 17 0 00 000441*                 GO      TRACE                           ; handle trace output
```

```

                                        ; 1st segment of test (Segment A)

000631' 400 15 0 00 000000      TA10:   SETZ    ERFLG,                  ; clear error flag
000632' 201 00 0 00 000001              MOVEI   1                       ; set subtest number
000633' 202 00 0 00 000471*             MOVEM   TSTSUB                  ;   to 1
000634' 260 17 0 00 000000*             GO      IPACLR                  ; do a 'port clear'
000635' 200 01 0 00 010734'             MOVE    1,[500003,,370467]      ; get data expected
000636' 202 01 0 00 010367'             MOVEM   1,SAVCS2                ; save it
000637' 201 01 0 00 377767              MOVEI   1,377767                ; get all 1's (except PCLEAR and MPRUN)
000640' 202 01 0 00 010366'             MOVEM   1,SAVCS1                ; save data written
000641' 260 17 0 00 000473*             GO      LDCSR                   ; write CSR
000642' 260 17 0 00 000474*             GO      RDCSR                   ; read CSR
000643' 474 15 0 00 000000              SETO    ERFLG,                  ; error reading CSR - flag it
000644' 202 01 0 00 010370'             MOVEM   1,SAVCS3                ; save data read
000645' 312 01 0 00 010367'             CAME    1,SAVCS2                ; data same as expected?
000646' 474 15 0 00 000000              SETO    ERFLG,                  ; no - flag error
000647' 027 00 0 00 000675'             SCOPER  MA10                    ; print error message
000650' 254 00 0 00 000631'             JRST    TA10                    ; loop on error
000651' 254 00 0 00 000674'             JRST    TX10                    ; altmode exit
000652' 326 15 0 00 000674'             JUMPN   ERFLG,TX10              ; error? yes - exit

                                        ; 2nd segment of test (Segment B)

000653' 400 15 0 00 000000      TB10:   SETZ    ERFLG,                  ; clear error flag
000654' 201 00 0 00 000002              MOVEI   2                       ; set up subtest number
000655' 202 00 0 00 000633*             MOVEM   TSTSUB                  ;   to 2
000656' 260 17 0 00 000634*             GO      IPACLR                  ; do a 'port clear'
000657' 200 01 0 00 010735'             MOVE    1,[500003,,000000]      ; get data expected
000660' 202 01 0 00 010367'             MOVEM   1,SAVCS2                ; save it
000661' 400 01 0 00 000000              SETZ    1,                      ; get all 0's
000662' 202 01 0 00 010366'             MOVEM   1,SAVCS1                ; save data written
000663' 260 17 0 00 000641*             GO      LDCSR                   ; write CSR
000664' 260 17 0 00 000642*             GO      RDCSR                   ; read CSR
000665' 474 15 0 00 000000              SETO    ERFLG,                  ; error reading CSR - flag it
000666' 202 01 0 00 010370'             MOVEM   1,SAVCS3                ; save data read
000667' 312 01 0 00 010367'             CAME    1,SAVCS2                ; data same as expected?
000670' 474 15 0 00 000000              SETO    ERFLG,                  ; no - flag error
000671' 027 00 0 00 000675'             SCOPER  MA10                    ; print error message
000672' 254 00 0 00 000653'             JRST    TB10                    ; loop on error
000673' 254 00 0 00 000674'             JRST    TX10                    ; altmode exit

                                        ; End of test

000674' 263 17 0 00 000000      TX10:   RTN                             ; return
```

```

                                        ; Error messages

000675' 240000  000677'         MA10:   CALL!TXNOT!MA10PN               ; print data written
000676' 270000  010254'                 LAST!CALL!TXALL!MPNT5           ; print CSR correct/actual

000677' 037 00 0 00 010706'     MA10PN: TMSGC   <Write CSR '>
000700' 200 00 0 00 010366'             MOVE    SAVCS1
000701' 037 13 0 00 000000              PNTHW
000702' 037 00 0 00 010711'             TMSG    <', then read CSR, verify result.>
000703' 263 17 0 00 000000              RTN
```

```

                                        ;#*****************************************************************
                                        ;* TEST 11 - EBUF Access Test
                                        ;
                                        ; Description:  This test is intended to do a simple test  to  if
                                        ;               it is possible to access the EBUF.  It just reads
                                        ;               it and if read it should be zero.
                                        ;
                                        ; Procedure:    EBUS Reset
                                        ;               Write CSR bit 19 a one (set 'DIAG Test EBUF')
                                        ;               Read EBUF
                                        ;               Data should be all 0's - if so - ok
                                        ;                                        if not - error
                                        ;
                                        ; Failure:      --
                                        ;#*****************************************************************

                                        ; Test data

000704' 254 00 0 00 000715'             TSTE11: JRST    TG11                    ; go start test
000705' 410000  000011                          EBUS!MBUS!ZEBUS!11              ; test mask
000706' 000000  010736'                         0,,[ASCIZ ^EBUF Access Test^]
000707' 010742' 010744'                         [EXP E20,MLAST!E9],,[EXP MLAST!E1]
000710' 000000  000735'                         TSTE12                          ; failure test table
000711' 000000  001106'                         TSTE14                          ;   ...
000712' 000000  001162'                         TSTE15
000713' 000000  001220'                         TSTE16
000714' 777777  777777                          -1

                                        ; Start test

000715' 201 00 0 00 000000'             TG11:   MOVEI   Z2                      ; get address of module start
000716' 260 17 0 00 000630*                     GO      TRACE                   ; handle trace output

000717' 400 15 0 00 000000              TA11:   SETZ    ERFLG,                  ; clear error flag
000720' 260 17 0 00 000453*                     GO      ERESET                  ; do an EBUS Reset
000721' 201 01 0 00 200000                      MOVEI   1,TSTEBF                ; set 'Diag Test Ebuf'
000722' 260 17 0 00 000663*                     GO      LDCSR                   ; write it
000723' 260 17 0 00 000000*                     GO      RDEBUF                  ; read EBUF
000724' 202 01 0 00 010362'                     MOVEM   1,SAVDA1                ; save data read
000725' 332 00 0 00 000001                      SKIPE   1                       ; get back zeros?
000726' 474 15 0 00 000000                      SETO    ERFLG,                  ; no - flag error
000727' 027 00 0 00 000733'                     SCOPER  MA11                    ; print error message
000730' 254 00 0 00 000717'                     JRST    TA11                    ; loop on error
000731' 254 00 0 00 000732'                     JRST    TX11                    ; altmode exit

                                        ; End of test

000732' 263 17 0 00 000000              TX11:   RTN                             ; return

                                        ; Error messages

000733' 140000  010745'                 MA11:   MSG!TXNOT![ASCIZ /Did EBUS Reset, then read EBUF./]
000734' 270000  010236'                         LAST!CALL!TXALL!MPNT4           ; print EBUS Correct/Actual
```

```
                                        
                                        ;#*********************************************************************
                                        ;* TEST 12 - EBUF/CSR Conflict Test
                                        ;
                                        ; Description:  This test verifies that writing the EBUF does not
                                        ;               also write the CSR Register.
                                        ;
                                        ; Procedure:    EBUS Reset
                                        ;               Write CSR 1's except 'Port Clear' and error bits.
                                        ;               Write 0's to EBUF (DATAO EBUF)
                                        ;               Read CSR
                                        ;               CSR bits already set should still be
                                        ;                 set - if so - ok
                                        ;                       if not - error
                                        ;
                                        ; Failure:      Any error implicates the RMUX which determines
                                        ;               which is read - CSR or EBUF.
                                        ;#*********************************************************************

                                        ; Test data

000735' 254 00 0 00 000745'             TSTE12: JRST    TG12                            ; go start test
000736' 410000  000012                          EBUS!MBUS!ZEBUS!12                      ; test mask
000737' 000000  010754'                         0,,[ASCIZ ^EBUF/CSR Conflict Test^]
000740' 010761' 010762'                         [EXP MLAST!E11],,[EXP E10,E1,E12,E13,E14,E15,E16,MLAST!E17]
000741' 000000  001106'                         TSTE14                                  ; failure test table
000742' 000000  001162'                         TSTE15                                  ;   ...
000743' 000000  001220'                         TSTE16
000744' 777777  777777                          -1

                                        ; Start test

000745' 201 00 0 00 000000'             TG12:   MOVEI   Z2                              ; get address of module start
000746' 260 17 0 00 000716*                     GO      TRACE                           ; handle trace output
000747' 200 00 0 00 010772'                     MOVE    [500003,,270467]                ; get correct data
000750' 202 00 0 00 010367'                     MOVEM   SAVCS2                          ; save it

000751' 400 15 0 00 000000              TA12:   SETZ    ERFLG,                          ; clear error flag
000752' 260 17 0 00 000720*                     GO      ERESET                          ; do an EBUS Reset
000753' 201 01 0 00 273767                      MOVEI   1,273767                        ; get CSR bits
000754' 260 17 0 00 000722*                     GO      LDCSR                           ; write to CSR
000755' 400 01 0 00 000000                      SETZ    1,                              ; clear AC1
000756' 260 17 0 00 000000*                     GO      LDEBUF                          ; write 0's to EBUF
000757' 260 17 0 00 000664*                     GO      RDCSR                           ; read CSR
000760' 474 15 0 00 000000                      SETO    ERFLG,                          ; error accessing CSR
000761' 202 01 0 00 010370'                     MOVEM   1,SAVCS3                        ; save CSR data
000762' 312 01 0 00 010367'                     CAME    1,SAVCS2                        ; same as expected?
000763' 474 15 0 00 000000                      SETO    ERFLG,                          ; no - flag error
000764' 027 00 0 00 000770'                     SCOPER  MA12                            ; print error message
000765' 254 00 0 00 000751'                     JRST    TA12                            ; loop on error
000766' 254 00 0 00 000767'                     JRST    TX12                            ; altmode exit
```

```

                                   ; End of test

000767' 263 17 0 00 000000         TX12:   RTN                                   ; return

                                   ; Error messages

000770' 140000  010773'            MA12:   MSG!TXNOT![ASCIZ /Wrote 273767 to CSR, wrote 0's to EBUF, then read CSR./]
000771' 140000  011006'                    MSG!TXNOT![ASCIZ /CSR should not have changed./]
000772' 270000  010254'                    LAST!CALL!TXALL!MPNT5                 ; print CSR correct/actual
```

```
                                        ;#**********************************************************************
                                        ;* TEST 13 - KMUX Parity Generators
                                        ;
                                        ; Description:  The port writes a series of data patterns to  the
                                        ;               KL10 all of which have good  parity.   If correct
                                        ;               parity was not generated, a page fail will occur.
                                        ;
                                        ; Procedure:    EBUS Reset
                                        ;               Load microcode
                                        ;               Start microcode at location 0
                                        ;
                                        ;               Write data pattern to the port
                                        ;               Read the same data pattern from the port
                                        ;               Did a page fail occur due to AR/ARX parity
                                        ;                 error?  if no - ok
                                        ;                         if yes - error
                                        ;
                                        ;               Repeat for each data pattern.
                                        ;
                                        ; Data Patterns: floating one's          777000,,777000
                                        ;               floating zero's          777777,,000000
                                        ;               floating 777's           000777,,777777
                                        ;               floating 000's           777000,,777777
                                        ;               000000,,777777           777777,,000777
                                        ;               000777,,000777           777777,,777000
                                        ;               777000,,000777           777777,,777777
                                        ;               000777,,777000           000000,,000000
                                        ;
                                        ; Failure:      KMUX parity generators
                                        ;#**********************************************************************

                                        ; Test data

000773' 254 00 0 00 001000'             TSTE13: JRST    TG13                    ; go start test
000774' 400400  000013                          EBUS!NDMP!ZEBUS!13              ; test mask
000775' 001063' 011014'                         T13M,,[ASCIZ ^KMUX Parity Generators^]
000776' 011021' 011022'                         [EXP MLAST!E18],,[EXP E9,MLAST!E22]
000777' 777777  777777                          -1                              ; failure test table

                                        ; Start test

001000' 332 00 0 00 030037              TG13:   SKIPE   USER                    ; user mode?
                                                GO      [MOVE   TSTFLG          ; yes - ensure this test does not
                                                         TLO    (TUSER)         ;     get run (but only if not in
                                                         SKIPN  UDEBUG          ;     debug mode)
                                                         MOVEM  TSTFLG
001001' 260 17 0 00 010511'                              RTN]
001002' 201 00 0 00 000000'                     MOVEI   Z2                      ; get address of module start
001003' 260 17 0 00 000746*                     GO      TRACE                   ; handle trace output
001004' 201 01 0 00 001063'                     MOVEI   1,T13M                  ; set up microcode address
001005' 260 17 0 00 000000*                     GO      TLOAD                   ; load/verify it
001006' 263 17 0 00 000000                      RTN                             ; failed - exit test
```

```
                                 ; 1st segment of test (Segment A) (Start the port)

001007' 400 15 0 00 000000       TA13:   SETZ    ERFLG,                  ; clear error flag
001010' 201 00 0 00 000001               MOVEI   1                       ; set subtest number
001011' 202 00 0 00 000655*              MOVEM   TSTSUB                  ;   to 1
001012' 201 00 0 00 000010               MOVEI   MPRUN                   ; only set MPRUN bit when
001013' 202 00 0 00 000000*              MOVEM   SDATA                   ;    starting the port
001014' 402 00 0 00 000000*              SETZM   SNEXT                   ; set start address to 0
001015' 260 17 0 00 010277'              GO      MSTART                  ; start up port
001016' 254 00 0 00 001007'              JRST    TA13                    ; loop on error
001017' 254 00 0 00 001045'              JRST    TX13                    ; altmode exit
001020' 326 15 0 00 001045'              JUMPN   ERFLG,TX13              ; error? yes - exit

                                 ; Remaining test segments

001021' 400 06 0 00 000000               SETZ    6,                      ; clear bit table pointer

                                 ; First, get the data pattern to write

001022' 400 15 0 00 000000       TB13:   SETZ    ERFLG,                  ; clear error flag
001023' 201 00 0 06 000002               MOVEI   2(6)                    ; set subtest number to
001024' 202 00 0 00 001011*              MOVEM   TSTSUB                  ;   pattern number + 1
001025' 200 01 0 06 001634'              MOVE    1,T23PAT(6)             ; get pattern to write
001026' 306 01 0 00 000377               CAIN    1,377                   ; end of table?
001027' 254 00 0 00 001045'              JRST    TX13                    ; yes - end of test
001030' 202  1 0 00 010362'              MOVEM   1,SAVDA1                ; save data

                                 ; Write/read data

001031' 402 00 0 00 000000*              SETZM   PFAIL                   ; clear page fail flag
001032' 260 17 0 00 000000*              GO      .DATAO                  ; write to port
001033' 260 17 0 00 000000*              GO      .DATAI                  ; read from port
001034' 202 01 0 00 010363'              MOVEM   1,SAVDA2                ; save data read
001035' 332 00 0 00 001031*              SKIPE   PFAIL                   ; ARX page fail occurred?
001036' 474 15 0 00 000000               SETO    ERFLG,                  ; yes - flag error

                                 ; Handle error printout

001037' 027 00 0 00 001046'              SCOPER  MA13                    ; print error message
001040' 254 00 0 00 001022'              JRST    TB13                    ; loop on error
001041' 254 00 0 00 001045'              JRST    TX13                    ; altmode exit
001042' 326 15 0 00 001045'              JUMPN   ERFLG,TX13              ; error? yes - exit

                                 ; Increment data pattern pointer

001043' 350 00 0 00 000006               AOS     6                       ; point to next data pattern
001044' 254 00 0 00 001022'              JRST    TB13                    ; repeat this segment

                                 ; End of test

001045' 263 17 0 00 000000       TX13:   RTN                             ; return
```

```

                                         ; Error messages

     001046' 240000  001050'             MA13:   CALL!TXNOT!MA13PN
     001047' 270000  010231'                     LAST!CALL!TXALL!MPNT2

     001050' 037 00 0 00 011024'         MA13PN: TMSGC   <Wrote '>
     001051' 200 00 0 00 010362'                 MOVE    SAVDA1
     001052' 037 13 0 00 000000                  PNTHW
     001053' 037 00 0 00 011026'                 TMSG    <' to port>
     001054' 037 00 0 00 011030'                 TMSGC   <Read '>
     001055' 200 00 0 00 010363'                 MOVE    SAVDA2
     001056' 037 13 0 00 000000                  PNTHW
     001057' 037 00 0 00 011032'                 TMSG    <' from port>
     001060' 037 00 0 00 011035'                 TMSGC   <Page fail (AR/ARX parity error) occurred>
     001061' 260 17 0 00 010314'                 GO      PFPNT                   ; print page fail data
     001062' 263 17 0 00 000000                  RTN

                                         ; Microcode

     001063' 000000  010000              T13M:   MWORD   <ADDR=0,JMAP,J=1>                       ; 0
     001064' 000000  000040
     001065' 000100  030000                      MWORD   <CJP,J=3,CENA,CCER>                     ; 1
     001066' 000400  100060
     001067' 000200  010000                      MWORD   <JMAP,J=1>                              ; 2
     001070' 000000  000040
     001071' 000300  100010                      MWORD   <LDCT,J=10,SD0,OR,D=2,SELE,MGC=10>      ; 3
     001072' 732000  005300
     001073' 000400  040000                      MWORD   <RPCT,J=4>                              ; 4
     001074' 000000  000220
     001075' 000500  070000                      MWORD   <CJP,J=7,CENA,CCER>                     ; 5
     001076' 000400  100060
     001077' 000600  050000                      MWORD   <JMAP,J=5>                              ; 6
     001100' 000000  000040
     001101' 000700  002020                      MWORD   <CONT,SOA,OR,D=1,OENA,SELE,MGC=20>      ; 7
     001102' 431000  005340
     001103' 001000  000000                      MWORD   <JZ>                                    ; 10
     001104' 000000  000000
     001105' 777777  777777                      -1
```

```
                                        ;#*********************************************************************
                                        ;* TEST 14 - EBUF/CSR Conflict Test
                                        ;
                                        ; Description:  This test does further verification that  writing
                                        ;               the EBUF does not also write the CSR Register.
                                        ;
                                        ; Procedure:    EBUS Reset
                                        ;               Write CSR setting bit 19
                                        ;               Write 1's to EBUF (DATAO EBUF)
                                        ;               Read CSR
                                        ;               CSR bits already set should still be
                                        ;                 set - if so - ok
                                        ;                       if not - error
                                        ;               Read EBUF (DATAI EBUF)
                                        ;               Data should be 1's - if so - ok
                                        ;                                    if not - error
                                        ;
                                        ; Failure:      Any error implicates the RMUX which determines
                                        ;               which is read - CSR or EBUF.
                                        ;#*********************************************************************

                                        ; Test data

001106' 254 00 0 00 001115'             TSTE14: JRST    TG14                    ; go start test
001107' 410000  000014                          EBUS!MBUS!ZEBUS!14              ; test mask
001110' 000000  010754'                         0,,[ASCIZ ^EBUF/CSR Conflict Test^]
001111' 010761' 010762'                         [EXP MLAST!E11],,[EXP E10,E1,E12,E13,E14,E15,E16,MLAST!E17]
001112' 000000  001162'                         TSTE15                          ; failure test table
001113' 000000  001220'                         TSTE16                          ;   ...
001114' 777777  777777                          -1

                                        ; Start test

001115' 201 00 0 00 000000'             TG14:   MOVEI   Z2                      ; get address of module start
001116' 260 17 0 00 001003*                     GO      TRACE                   ; handle trace output
001117' 200 00 0 00 011046'                     MOVE    [500003,,200000]        ; get correct CSR data
001120' 202 00 0 00 010367'                     MOVEM   SAVCS2                  ; save it
001121' 476 00 0 00 010372'                     SETOM   SAVEB1                  ; set up correct EBUF data
```

```

                                        ; 1st segment of test (Segment A)

001122' 400 15 0 00 000000      TA14:   SETZ    ERFLG,                  ; clear error flag
001123' 201 00 0 00 000001              MOVEI   1                       ; set subtest number
001124' 202 00 0 00 001024*             MOVEM   TSTSUB                  ;   to 1
001125' 260 17 0 00 000752*             GO      ERESET                  ; do an EBUS Reset
001126' 201 01 0 00 200000              MOVEI   1,TSTEBF                ; get 'DIAG Test EBUF' bit
001127' 260 17 0 00 000754*             GO      LDCSR                   ; write to CSR register
001130' 474 01 0 00 000000              SETO    1,                      ; get 1's
001131' 260 17 0 00 000756*             GO      LDEBUF                  ; write to EBUF
001132' 260 17 0 00 000757*             GO      RDCSR                   ; read CSR
001133' 474 15 0 00 000000              SETO    ERFLG,                  ; error accessing CSR
001134' 202 01 0 00 010370'             MOVEM   1,SAVCS3                ; save CSR data
001135' 312 01 0 00 010367'             CAME    1,SAVCS2                ; CSR correct?
001136' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
001137' 027 00 0 00 001154'             SCOPER  MA14                    ; print error message
001140' 254 00 0 00 001122'             JRST    TA14                    ; loop on error
001141' 254 00 0 00 001153'             JRST    TX14                    ; altmode exit
001142' 326 15 0 00 001153'             JUMPN   ERFLG,TX14              ; error? yes - exit

                                        ; 2nd segment of test (Segment B)

001143' 350 00 0 00 001124*             AOS     TSTSUB                  ; point to next subtest
001144' 260 17 0 00 000723*             GO      RDEBUF                  ; read EBUF
001145' 202 01 0 00 010373'             MOVEM   1,SAVEB2                ; save EBUF data
001146' 312 01 0 00 010372'             CAME    1,SAVEB1                ; EBUF data correct?
001147' 474 15 0 00 000000              SETO    ERFLG,                  ; no - flag error
001150' 027 00 0 00 001157'             SCOPER  MB14                    ; print error message
001151' 254 00 0 00 001122'             JRST    TA14                    ; loop on error
001152' 254 00 0 00 001153'             JRST    TX14                    ; altmode exit

                                        ; End of test

001153' 263 17 0 00 000000      TX14:   RTN                             ; return

                                        ; Error messages

001154' 140000  011047'         MA14:   MSG!TXNOT![ASCIZ /Write CSR 'Diag Test Ebuf', write 1's to EBUF.  CSR got/]
001155' 140000  011063'                 MSG!TXNOT![ASCIZ /written by write of EBUF./]
001156' 270000  010254'                 LAST!CALL!TXALL!MPNT5           ; print CSR correct/actual

001157' 140000  011071'         MB14:   MSG!TXNOT![ASCIZ /Write CSR 'Diag Test Ebuf', write 1's to EBUF/]
001160' 140000  011103'                 MSG!TXNOT![ASCIZ /Read CSR, then read EBUF.  EBUF incorrect/]
001161' 270000  010236'                 LAST!CALL!TXALL!MPNT4           ; print EBUF correct/actual
```

```

                                      ;#*********************************************************************
                                      ;* TEST 15 - EBUF Reset Test
                                      ;
                                      ; Description:  This test verifies that the EBUF is cleared by an
                                      ;               EBUS Reset.
                                      ;
                                      ; Procedure:    EBUS Reset
                                      ;               Write CSR setting bit 19
                                      ;               Write 1's to EBUF (DATAO EBUF)
                                      ;               EBUS Reset
                                      ;               Write CSR setting bit 19
                                      ;               Read EBUF (DATAI EBUF)
                                      ;               Data should be 0's - if so - ok
                                      ;                                        if not - error
                                      ;
                                      ; Failure:      Any error implicates reset logic or EBUF.
                                      ;#*********************************************************************

                                      ; Test data

001162' 254 00 0 00 001170'           TSTE15: JRST    TG15                    ; go start test
001163' 400000  000015                        EBUS!ZEBUS!15                   ; test mask
001164' 000000  011114'                       0,,[ASCIZ ^EBUF Reset Test^]
001165' 011120' 000000                        [EXP E6,MLAST!E20],,0
001166' 000000  001220'                       TSTE16                          ; failure test table
001167' 777777  777777                        -1                              ;  ...

                                      ; Start test

001170' 332 00 0 00 030037            TG15:   SKIPE   USER                    ; user mode?
                                              GO      [MOVE   TSTFLG          ; yes - ensure this test does not
                                                       TLO    (TUSER)         ;    get run (but only if not in
                                                       SKIPN  UDEBUG          ;    debug mode)
                                                       MOVEM  TSTFLG
001171' 260 17 0 00 010511'                            RTN]
001172' 201 00 0 00 000000'                   MOVEI   Z2                      ; get address of module start
001173' 260 17 0 00 001116*                   GO      TRACE                   ; handle trace output
001174' 402 00 0 00 010372'                   SETZM   SAVEB1                  ; set up correct EBUF data
```

```

001175' 400 15 0 00 000000      TA15:   SETZ    ERFLG,                  ; clear error flag
001176' 260 17 0 00 001125*             GO      ERESET                  ; do an EBUS Reset
001177' 201 01 0 00 200000              MOVEI   1,TSTEBF                ; get 'DIAG Test EBUF' bit
001200' 260 17 0 00 001127*             GO      LDCSR                   ; write to CSR
001201' 474 01 0 00 000000              SETO    1,                      ; get all 1's
001202' 260 17 0 00 001131*             GO      LDEBUF                  ; write to EBUF
001203' 260 17 0 00 001176*             GO      ERESET                  ; do an EBUS Reset
001204' 201 01 0 00 200000              MOVEI   1,TSTEBF                ; get 'DIAG Test EBUF' bit
001205' 260 17 0 00 001200*             GO      LDCSR                   ; write to CSR
001206' 260 17 0 00 001144*             GO      RDEBUF                  ; write to EBUF
001207' 202 01 0 00 010373'             MOVEM   1,SAVEB2                ; save EBUF data
001210' 312 01 0 00 010372'             CAME    1,SAVEB1                ; correct (zero)?
001211' 474 15 0 00 000000              SETO    ERFLG,                  ; no - flag error
001212' 027 00 0 00 001216'             SCOPER  MA15                    ; print error message
001213' 254 00 0 00 001175'             JRST    TA15                    ; loop on error
001214' 254 00 0 00 001215'             JRST    TX15                    ; altmode exit

                                ; End of test

001215' 263 17 0 00 000000      TX15:   RTN                             ; return

                                ; Error messages

001216' 140000  011122'         MA15:   MSG!TXNOT![ASCIZ /EBUS Reset did not clear EBUF./]
001217' 270000  010236'                 LAST!CALL!TXALL!MPNT4           ; print EBUF correct/actual
```

```

                                   ;#**********************************************************************
                                   ;* TEST 16 - EBUF Data Path Test
                                   ;
                                   ; Description:  This test verifies that the data path to the EBUF
                                   ;               and looped back on the MBus is valid.
                                   ;
                                   ; Procedure:    EBUS Reset
                                   ;               Write CSR setting bit 19
                                   ;               Write data pattern to EBUF (DATAO EBUF)
                                   ;               Read EBUF (DATAI EBUF)
                                   ;               Data the same - if so - ok
                                   ;                                if not - error
                                   ;
                                   ;               Repeat with remaining data patterns
                                   ;
                                   ; Data Patterns: zero's
                                   ;               one's
                                   ;               alternating (52..)
                                   ;               alternating (25..)
                                   ;               floating one
                                   ;               floating zero
                                   ;
                                   ; Failure:      ---
                                   ;#**********************************************************************

                                   ; Test data

001220' 254 00 0 00 001225'        TSTE16: JRST    TG16                    ; go start test
001221' 410000  000016                     EBUS!MBUS!ZEBUS!16              ; test mask
001222' 000000  011131'                    0,,[ASCIZ ^EBUF Data Path Test^]
001223' 011121' 011135'                    [EXP MLAST!E20],,[EXP MLAST!E6]
001224' 777777  777777                     -1                              ; failure test table

                                   ; Start test

001225' 201 00 0 00 000000'        TG16:   MOVEI   Z2                      ; get address of module start
001226' 260 17 0 00 001173*                GO      TRACE                   ; handle trace output
001227' 400 06 0 00 000000                 SETZ    6,                      ; clear bit table pointer

                                   ; First, get the data pattern to write

001230' 400 15 0 00 000000         TA16:   SETZ    ERFLG,                  ; clear error flag
001231' 201 00 0 06 000001                 MOVEI   1(6)                    ; set subtest number to
001232' 202 00 0 00 001143*                MOVEM   TSTSUB                  ;   pattern number + 1
001233' 200 01 0 06 001261'                MOVE    1,T16PAT(6)             ; get pattern to write
001234' 306 01 0 00 000377                 CAIN    1,377                   ; end of table?
001235' 254 00 0 00 001256'                JRST    TX16                    ; yes - end of test
```

```

                                        ; First test segment - verify EBUF write/read correctly

001236' 202 01 0 00 010372'                     MOVEM   1,SAVEB1                ; save data
001237' 260 17 0 00 001203*                     GO      ERESET                  ; do an EBUS Reset
001240' 201 01 0 00 200000                      MOVEI   1,TSTEBF                ; get 'DIAG Test EBUF' bit
001241' 260 17 0 00 001205*                     GO      LDCSR                   ; write to CSR
001242' 200 01 0 00 010372'                     MOVE    1,SAVEB1                ; get data pattern
001243' 260 17 0 00 001202*                     GO      LDEBUF                  ; write to EBUF
001244' 260 17 0 00 001206*                     GO      RDEBUF                  ; write to EBUF
001245' 202 01 0 00 010373'                     MOVEM   1,SAVEB2                ; save EBUF data
001246' 312 01 0 00 010372'                     CAME    1,SAVEB1                ; correct?
001247' 474 15 0 00 000000                      SETO    ERFLG,                  ; no - flag error
001250' 027 00 0 00 001257'                     SCOPER  MA16                    ; print error message
001251' 254 00 0 00 001230'                     JRST    TA16                    ; loop on error
001252' 254 00 0 00 001256'                     JRST    TX16                    ; altmode exit
001253' 326 15 0 00 001256'                     JUMPN   ERFLG,TX16              ; error? yes - exit

                                        ; Increment data pattern pointer

001254' 350 00 0 00 000006                      AOS     6                       ; point to next data pattern
001255' 254 00 0 00 001230'                     JRST    TA16                    ; repeat this segment

                                        ; End of test

001256' 263 17 0 00 000000              TX16:   RTN                     ; return

                                        ; Error messages

001257' 140000  011136'                 MA16:   MSG!TXNOT![ASCIZ /EBUF data loopback failed./]
001260' 270000  010236'                         LAST!CALL!TXALL!MPNT4           ; print EBUF correct/actual

                                        ; Data Patterns

001261' 000000  000000                  T16PAT: EXP     0,-2,525252525252,252525252525
001262' 777777  777776
001263' 525252  525252
001264' 252525  252525
001265' 000000  000001                          EXP     000000000001,000000000002,000000000004
001266' 000000  000002
001267' 000000  000004
001270' 000000  000010                          EXP     000000000010,000000000020,000000000040
001271' 000000  000020
001272' 000000  000040
001273' 000000  000100                          EXP     000000000100,000000000200,000000000400
001274' 000000  000200
001275' 000000  000400
001276' 000000  001000                          EXP     000000001000,000000002000,000000004000
001277' 000000  002000
001300' 000000  004000
001301' 000000  010000                          EXP     000000010000,000000020000,000000040000
001302' 000000  020000
001303' 000000  040000
001304' 000000  100000                          EXP     000000100000,000000200000,000000400000
001305' 000000  200000
```

```
001306' 000000  400000
001307' 000001  000000                  EXP     000001000000,000002000000,000004000000
001310' 000002  000000
001311' 000004  000000
001312' 000010  000000                  EXP     000010000000,000020000000,000040000000
001313' 000020  000000
001314' 000040  000000
001315' 000100  000000                  EXP     000100000000,000200000000,000400000000
001316' 000200  000000
001317' 000400  000000
001320' 001000  000000                  EXP     001000000000,002000000000,004000000000
001321' 002000  000000
001322' 004000  000000
001323' 010000  000000                  EXP     010000000000,020000000000,040000000000
001324' 020000  000000
001325' 040000  000000
001326' 100000  000000                  EXP     100000000000,200000000000,400000000000
001327' 200000  000000
001330' 400000  000000
001331' 777777  777776                  EXP     777777777776,777777777775,777777777773
001332' 777777  777775
001333' 777777  777773
001334' 777777  777767                  EXP     777777777767,777777777757,777777777737
001335' 777777  777757
001336' 777777  777737
001337' 777777  777677                  EXP     777777777677,777777777577,777777777377
001340' 777777  777577
001341' 777777  777377
001342' 777777  776777                  EXP     777777776777,777777775777,777777773777
001343' 777777  775777
001344' 777777  773777
001345' 777777  767777                  EXP     777777767777,777777757777,777777737777
001346' 777777  757777
001347' 777777  737777
001350' 777777  677777                  EXP     777777677777,777777577777,777777377777
001351' 777777  577777
001352' 777777  377777
001353' 777776  777777                  EXP     777776777777,777775777777,777773777777
001354' 777775  777777
001355' 777773  777777
001356' 777767  777777                  EXP     777767777777,777757777777,777737777777
001357' 777757  777777
001360' 777737  777777
001361' 777677  777777                  EXP     777677777777,777577777777,777377777777
001362' 777577  777777
001363' 777377  777777
001364' 776777  777777                  EXP     776777777777,775777777777,773777777777
001365' 775777  777777
001366' 773777  777777
001367' 767777  777777                  EXP     767777777777,757777777777,737777777777
001370' 757777  777777
001371' 737777  777777
001372' 677777  777777                  EXP     677777777777,577777777777,377777777777
001373' 577777  777777
001374' 377777  777777
```

```
  1397  001375' 000000  000377                  EXP     377                     ; table terminator
```

```
                                        ;#*******************************************************************
                                        ;* TEST 17 - CSR Bit 18 Self Clearing Test
                                        ;
                                        ; Description:  This test verifies  that  the  clear  bit  resets
                                        ;               after the clear is done.
                                        ;
                                        ; Procedure:    EBUS Reset
                                        ;               Write CSR setting bit 18
                                        ;               Read CSR
                                        ;               Bit 18 cleared?  if so - ok
                                        ;                                if not - error
                                        ;
                                        ; Failure:      ---
                                        ;#*******************************************************************

                                        ; Test data

001376' 254 00 0 00 001405'             TSTE17: JRST    TG17                    ; go start test
001377' 400000  000017                          EBUS!ZEBUS!17                   ; test mask
001400' 000000  011144'                         0,,[ASCIZ ^CSR Bit 18 Self Clearing Test^]
001401' 011135' 000000                          [EXP MLAST!E6],,0
001402' 000000  001424'                         TSTE20                          ; failure test table
001403' 000000  001453'                         TSTE21                          ;   ...
001404' 777777  777777                          -1

                                        ; Start test

001405' 201 00 0 00 000000'             TG17:   MOVEI   Z2                      ; get address of module start
001406' 260 17 0 00 001226*                     GO      TRACE                   ; handle trace output

001407' 400 15 0 00 000000              TA17:   SETZ    ERFLG,                  ; clear error flag
001410' 260 17 0 00 000656*                     GO      IPACLR                  ; do a 'port clear'
001411' 260 17 0 00 001132*                     GO      RDCSR                   ; read CSR
001412' 474 15 0 00 000000                      SETO    ERFLG,                  ; error accessing CSR
001413' 202 01 0 00 010365'                     MOVEM   1,SAVCSR                ; save data
001414' 602 01 0 00 400000                      TRNE    1,PCLEAR                ; 'Port Clear' still set?
001415' 474 15 0 00 000000                      SETO    ERFLG,                  ; yes - flag error
001416' 027 00 0 00 001422'                     SCOPER  MA17                    ; print error message
001417' 254 00 0 00 001407'                     JRST    TA17                    ; loop on error
001420' 254 00 0 00 001421'                     JRST    TX17                    ; altmode exit

                                        ; End of test

001421' 263 17 0 00 000000              TX17:   RTN                             ; return

                                        ; Error messages

001422' 140000  011152'                 MA17:   MSG!TXNOT![ASCIZ /CSR Bit 18 'Port Clear' did not self-clear./]
001423' 270000  010231'                         LAST!CALL!TXALL!MPNT2           ; print CSR data
```

```

                                        ;#*********************************************************************
                                        ;* TEST 20 - CSR Bit 18 Reset Test
                                        ;
                                        ; Description:  This test verifies that the clear bit resets  the
                                        ;               port module (in particular the CSR register).
                                        ;
                                        ; Procedure:    EBUS Reset
                                        ;               Write CSR setting bit 18 and other bits
                                        ;               Read CSR
                                        ;               Bits cleared?  if so - ok
                                        ;                              if not - error
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

001424' 254 00 0 00 001432'             TSTE20: JRST    TG20                    ; go start test
001425' 400000  000020                          EBUS!ZEBUS!20                   ; test mask
001426' 000000  011163'                         0,,[ASCIZ ^CSR Bit 18 Reset Test^]
001427' 011170' 010744'                         [EXP E6,E11,E12,E13,E14,E15,E16,MLAST!E17],,[EXP MLAST!E1]
001430' 000000  001453'                         TSTE21                          ; failure test table
001431' 777777  777777                          -1                              ;   ...

                                        ; Start test

001432' 201 00 0 00 000000'             TG20:   MOVEI   Z2                      ; get address of module start
001433' 260 17 0 00 001406*                     GO      TRACE                   ; handle trace output
001434' 200 00 0 00 010653'                     MOVE    [400003,,0]             ; get correct CSR data
001435' 202 00 0 00 010367'                     MOVEM   SAVCS2                  ; save it

001436' 400 15 0 00 000000              TA20:   SETZ    ERFLG,                  ; clear error flag
001437' 260 17 0 00 001410*                     GO      IPACLR                  ; do a 'port clear'
001440' 260 17 0 00 001411*                     GO      RDCSR                   ; read CSR
001441' 474 15 0 00 000000                      SETO    ERFLG,                  ; error accessing CSR
001442' 202 01 0 00 010370'                     MOVEM   1,SAVCS3                ; save data
001443' 312 01 0 00 010367'                     CAME    1,SAVCS2                ; CSR cleared?
001444' 474 15 0 00 000000                      SETO    ERFLG,                  ; no - flag error
001445' 027 00 0 00 001451'                     SCOPER  MA20                    ; print error message
001446' 254 00 0 00 001436'                     JRST    TA20                    ; loop on error
001447' 254 00 0 00 001450'                     JRST    TX20                    ; altmode exit

                                        ; End of test

001450' 263 17 0 00 000000              TX20:   RTN                             ; return

                                        ; Error messages

001451' 140000  011200'                 MA20:   MSG!TXNOT![ASCIZ /CSR Bit 18 'Port Clear' did not reset proper bits in CSR./]
001452' 270000  010254'                         LAST!CALL!TXALL!MPNT5           ; print CSR correct/actual
```

```

                                                ;#*****************************************************************
                                                ;* TEST 21 - CSR Bit 18 Reset Test
                                                ;
                                                ; Description:  This test verifies that the clear bit resets  the
                                                ;               port module (in particular the EBUF).
                                                ;
                                                ; Procedure:    EBUS Reset
                                                ;               Write CSR setting bit 19
                                                ;               Write 1's to EBUF (DATAO EBUF)
                                                ;               Write CSR setting bit 18
                                                ;               Write CSR setting bit 19
                                                ;               Read EBUF (DATAI EBUF)
                                                ;               Bits cleared?  if so - ok
                                                ;                              if not - error
                                                ;
                                                ; Failure:      ---
                                                ;#*****************************************************************

                                                ; Test data

    001453' 254 00 0 00 001460'                 TSTE21: JRST    TG21                    ; go start test
    001454' 400000  000021                              EBUS!ZEBUS!21                   ; test mask
    001455' 000000  011163'                             0,,[ASCIZ ^CSR Bit 18 Reset Test^]
    001456' 011120' 000000                              [EXP E6,MLAST!E20],,0
    001457' 777777  777777                              -1                              ; failure test table

                                                ; Start test

    001460' 201 00 0 00 000000'                 TG21:   MOVEI   Z2                      ; get address of module start
    001461' 260 17 0 00 001433*                         GO      TRACE                   ; handle trace output
    001462' 402 00 0 00 010372'                         SETZM   SAVEB1                  ; set up EBUF correct data

    001463' 400 15 0 00 000000                  TA21:   SETZ    ERFLG,                  ; clear error flag
    001464' 260 17 0 00 001237*                         GO      ERESET                  ; do an EBUS Reset
    001465' 201 01 0 00 200000                          MOVEI   1,TSTEBF                ; get 'DIAG Test EBUF' bit
    001466' 260 17 0 00 001241*                         GO      LDCSR                   ; write to CSR
    001467' 474 01 0 00 000000                          SETO    1,                      ; get all 1's
    001470' 260 17 0 00 001243*                         GO      LDEBUF                  ; write to EBUF
    001471' 260 17 0 00 001437*                         GO      IPACLR                  ; do a 'Port Clear'
    001472' 201 01 0 00 200000                          MOVEI   1,TSTEBF                ; get 'DIAG Test EBUF' bit
    001473' 260 17 0 00 001466*                         GO      LDCSR                   ; write to CSR
    001474' 260 17 0 00 001244*                         GO      RDEBUF                  ; write to EBUF
    001475' 202 01 0 00 010373'                         MOVEM   1,SAVEB2                ; save data
    001476' 332 00 0 00 000001                          SKIPE   1                       ; zero?
    001477' 474 15 0 00 000000                          SETO    ERFLG,                  ; no - flag error
    001500' 027 00 0 00 001504'                         SCOPER  MA21                    ; print error message
    001501' 254 00 0 00 001463'                         JRST    TA21                    ; loop on error
    001502' 254 00 0 00 001503'                         JRST    TX21                    ; altmode exit
```

```
 1548
 1549                                    ; End of test
 1550
 1551   001503' 263 17 0 00 000000      TX21:   RTN                             ; return
 1552
 1553                                    ; Error messages
 1554
 1555   001504' 140000  011214'         MA21:   MSG!TXNOT![ASCIZ /'Port Clear' did not clear EBUF./]
 1556   001505' 270000  010236'                 LAST!CALL!TXALL!MPNT4           ; print EBUF correct/actual
```

```


                                                     ;#*******************************************************************
                                                     ;* TEST 22 - Read/Write EBUF When Ucode Running
                                                     ;
                                                     ; Description:   Verify that the EBUF cannot be written  from  the
                                                     ;                KL if the microprocessor is running.
                                                     ;
                                                     ; Procedure:     KL> EBUS Reset
                                                     ;                KL> Load test ucode
                                                     ;                KL> Set CSR Bit 19 (DIAG TEST EBUF) and clear EBUF
                                                     ;                KL> Start micro-sequencer at location 0
                                                     ;                KL> Read CSR and verify that MPRUN is set
                                                     ;                KL> Write data pattern to EBUF, read EBUF, Verify
                                                     ;                        that nothing was read.
                                                     ;                KL> Repeat with several data patterns:
                                                     ;                                00..01
                                                     ;                                one's
                                                     ;                                5252..
                                                     ;                                2525..
                                                     ;
                                                     ; Failure:       The EBUF can be read from the KL side only if the
                                                     ;                port is not running.  This is so the KL does  not
                                                     ;                inadvertently destroy the microcode, while  it is
                                                     ;                running.
                                                     ;#*******************************************************************

                                                     ; Test data

 001506' 254 00 0 00 001515'                         TSTE22: JRST    TG22                            ; go start test
 001507' 400400  000022                                      EBUS!NDMP!ZEBUS!22                      ; test mask
 001510' 001561' 011223'                                     T22M,,[ASCIZ ^Read/Write EBUF When Ucode Running^]
 001511' 010744' 011232'                                     [EXP MLAST!E1],,[EXP E10,E9,MLAST!E20]
 001512' 000000  004751'                                     TSTE55                                  ; failure test table
 001513' 000000  005025'                                     TSTE56                                  ;   ...
 001514' 777777  777777                                      -1

                                                     ; Start test

 001515' 201 00 0 00 000000'                         TG22:   MOVEI   Z2                              ; get address of module start
 001516' 260 17 0 00 001461*                                 GO      TRACE                           ; handle trace output
 001517' 201 01 0 00 001561'                                 MOVEI   1,T22M                          ; set up microcode address
 001520' 260 17 0 00 001005*                                 GO      TLOAD                           ; load/verify it
 001521' 263 17 0 00 000000                                  RTN                                     ; failed - exit test
```

```

                                    ; 1st segment of test (Segment A) - Start up port

001522' 400 15 0 00 000000          TA22:   SETZ    ERFLG,                          ; clear error flag
001523' 201 00 0 00 000001                  MOVEI   1                               ; set subtest number
001524' 202 00 0 00 001232*                 MOVEM   TSTSUB                          ;   to 1
001525' 260 17 0 00 001471*                 GO      IPACLR                          ; do a 'port clear'
001526' 201 01 0 00 200000                  MOVEI   1,TSTEBF                        ; get 'Diag Test EBUF' bit
001527' 202 01 0 00 001013*                 MOVEM   1,SDATA                         ; save it for start routine
001530' 260 17 0 00 001473*                 GO      LDCSR                           ; set it in CSR Register
001531' 400 01 0 00 000000                  SETZ    1,                              ; clear AC1
001532' 260 17 0 00 001470*                 GO      LDEBUF                          ; clear EBUF
001533' 402 00 0 00 001014*                 SETZM   SNEXT                           ; set start address to 0
001534' 260 17 0 00 010277'                 GO      MSTART                          ; start up port
001535' 254 00 0 00 001522'                 JRST    TA22                            ; loop on error
001536' 254 00 0 00 001556'                 JRST    TX22                            ; altmode exit
001537' 326 15 0 00 001556'                 JUMPN   ERFLG,TX22                      ; error? yes - exit

                                    ; 2nd segments (B) - Try to write data to EBUF

001540' 402 00 0 00 010372'                 SETZM   SAVEB1                          ; set up valid EBUF data
001541' 201 02 0 00 000003                  MOVEI   2,3                             ; count of data patterns
001542' 350 00 0 00 001524*         TB22:   AOS     TSTSUB                          ; point to next subtest
                                            MOVE    1,[1                            ; get data to write
                                                      -1
                                                       525252525252
001543' 200 01 0 02 011235'                            252525252525](2)
001544' 260 17 0 00 001532*                 GO      LDEBUF                          ; write EBUF
001545' 260 17 0 00 001474*                 GO      RDEBUF                          ; read EBUF
001546' 332 00 0 00 000001                  SKIPE   1                               ; data zero (failed)?
001547' 474 15 0 00 000000                  SETO    ERFLG,                          ; no - flag error
001550' 202 01 0 00 010373'                 MOVEM   1,SAVEB2                        ; save data
001551' 027 00 0 00 001557'                 SCOPER  MB22                            ; print error message
001552' 254 00 0 00 001522'                 JRST    TA22                            ; loop on error
001553' 254 00 0 00 001556'                 JRST    TX22                            ; altmode exit
001554' 326 15 0 00 001556'                 JUMPN   ERFLG,TX22                      ; abort test on first error
001555' 365 02 0 00 001542'                 SOJGE   2,TB22                          ; loop on data patterns

                                    ; End of test

001556' 263 17 0 00 000000          TX22:   RTN                                     ; return

                                    ; Error message - EBUF read and it shouldn't have been

001557' 140000  011241'             MB22:   MSG!TXNOT![ASCIZ /Shouldn't have been able to read EBUF/]
001560' 270000  010236'                     LAST!CALL!TXALL!MPNT4

                                    ; Microcode:

001561' 000000  000000              T22M:   MWORD   <ADDR=0,JZ>
001562' 000000  000000
001563' 777777  777777                      -1
```

```
                                        ;#*********************************************************************
                                        ;* TEST 23 - Par Gen - Not Forcing an Error
                                        ;
                                        ; Description:  Write a series of data patterns  to  verify  that
                                        ;               correct  parity  is  being  checked when  data is
                                        ;               received over the EBUS.
                                        ;
                                        ; Procedure:    EBUS Reset
                                        ;               Write CSR Bit 19 (DIAG TEST EBUF)
                                        ;               Write data pattern to EBUF
                                        ;               Read CSR register
                                        ;               Did parity error bit set? if so - error
                                        ;                                         if not - ok
                                        ;
                                        ;               Repeat for each data pattern.
                                        ;
                                        ; Data Patterns: floating one's        777000,,777000
                                        ;               floating zero's        777777,,000000
                                        ;               floating 777's         000777,,777777
                                        ;               floating 000's         777777,,777000
                                        ;               000000,,777777         777777,,000777
                                        ;               000777,,000777         777777,,777000
                                        ;               777000,,000777         777777,,777777
                                        ;               000777,,777000         000000,,000000
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

 001564' 254 00 0 00 001572'            TSTE23: JRST    TG23                    ; go start test
 001565' 400000  000023                         EBUS!ZEBUS!23                   ; test mask
 001566' 000000  011251'                        0,,[ASCIZ ^Par Gen - Not Forcing an Error^]
 001567' 011260' 011262'                        [EXP E18,MLAST!E19],,[EXP E16,MLAST!E20]
 001570' 000000  001773'                        TSTE24                          ; failure test table
 001571' 777777  777777                         -1                              ;   ...

                                        ; Start test

 001572' 201 00 0 00 000000'            TG23:   MOVEI   Z2                      ; get address of module start
 001573' 260 17 0 00 001516*                    GO      TRACE                   ; handle trace output
 001574' 400 06 0 00 000000                     SETZ    6,                      ; clear bit table pointer
```

```

                                        ; First, get the data pattern to write

001575' 400 15 0 00 000000      TA23:   SETZ    ERFLG,                  ; clear error flag
001576' 201 00 0 06 000001              MOVEI   1(6)                    ; set subtest number to
001577' 202 00 0 00 001542*             MOVEM   TSTSUB                  ;   pattern number + 1
001600' 200 01 0 06 001634'             MOVE    1,T23PAT(6)             ; get pattern to write
001601' 306 01 0 00 000377              CAIN    1,377                   ; end of table?
001602' 254 00 0 00 001624'             JRST    TX23                    ; yes - end of test
001603' 202 01 0 00 010361'             MOVEM   1,SAVDAT                ; save data
001604' 260 17 0 00 001464*             GO      ERESET                  ; do an EBUS Reset
001605' 201 01 0 00 200000              MOVEI   1,TSTEBF                ; get 'DIAG Test EBUF' bit
001606' 260 17 0 00 001530*             GO      LDCSR                   ; write to CSR
001607' 200 01 0 00 010361'             MOVE    1,SAVDAT                ; get data pattern
001610' 260 17 0 00 001544*             GO      LDEBUF                  ; write to EBUF
001611' 260 17 0 00 001440*             GO      RDCSR                   ; write to EBUF
001612' 474 15 0 00 000000              SETO    ERFLG,                  ; error accessing CSR
001613' 202 01 0 00 010365'             MOVEM   1,SAVCSR                ; save CSR data
001614' 602 01 0 00 004000              TRNE    1,EBUSPE                ; parity error set?
001615' 474 15 0 00 000000              SETO    ERFLG,                  ; yes - flag error
001616' 027 00 0 00 001625'             SCOPER  MA23                    ; print error message
001617' 254 00 0 00 001575'             JRST    TA23                    ; loop on error
001620' 254 00 0 00 001624'             JRST    TX23                    ; altmode exit
001621' 326 15 0 00 001624'             JUMPN   ERFLG,TX23              ; error? yes - exit

                                        ; Increment data pattern pointer

001622' 350 00 0 00 000006              AOS     6                       ; point to next data pattern
001623' 254 00 0 00 001575'             JRST    TA23                    ; repeat this segment

                                        ; End of test

001624' 263 17 0 00 000000      TX23:   RTN                     ; return

                                        ; Error messages

001625' 240000  001627'         MA23:   CALL!TXNOT!MA23PN
001626' 270000  010272'                 LAST!CALL!TXALL!MPNT6

001627' 037 00 0 00 011024'     MA23PN: TMSGC   <Wrote '>
001630' 200 00 0 00 010361'             MOVE    SAVDAT
001631' 037 13 0 00 000000              PNTHW
001632' 037 00 0 00 011264'             TMSG    <' to EBUF - PE occurred.>
001633' 263 17 0 00 000000              RTN
```

```


                                        ; Data Patterns

001634' 000000  000001                  T23PAT: EXP     000000000001,000000000002,000000000004
001635' 000000  000002
001636' 000000  000004
001637' 000000  000010                          EXP     000000000010,000000000020,000000000040
001640' 000000  000020
001641' 000000  000040
001642' 000000  000100                          EXP     000000000100,000000000200,000000000400
001643' 000000  000200
001644' 000000  000400
001645' 000000  001000                          EXP     000000001000,000000002000,000000004000
001646' 000000  002000
001647' 000000  004000
001650' 000000  010000                          EXP     000000010000,000000020000,000000040000
001651' 000000  020000
001652' 000000  040000
001653' 000000  100000                          EXP     000000100000,000000200000,000000400000
001654' 000000  200000
001655' 000000  400000
001656' 000001  000000                          EXP     000001000000,000002000000,000004000000
001657' 000002  000000
001660' 000004  000000
001661' 000010  000000                          EXP     000010000000,000020000000,000040000000
001662' 000020  000000
001663' 000040  000000
001664' 000100  000000                          EXP     000100000000,000200000000,000400000000
001665' 000200  000000
001666' 000400  000000
001667' 001000  000000                          EXP     001000000000,002000000000,004000000000
001670' 002000  000000
001671' 004000  000000
001672' 010000  000000                          EXP     010000000000,020000000000,040000000000
001673' 020000  000000
001674' 040000  000000
001675' 100000  000000                          EXP     100000000000,200000000000,400000000000
001676' 200000  000000
001677' 400000  000000
001700' 777777  777776                          EXP     777777777776,777777777775,777777777773
001701' 777777  777775
001702' 777777  777773
001703' 777777  777767                          EXP     777777777767,777777777757,777777777737
001704' 777777  777757
001705' 777777  777737
001706' 777777  777677                          EXP     777777777677,777777777577,777777777377
001707' 777777  777577
001710' 777777  777377
001711' 777777  776777                          EXP     777777776777,777777775777,777777773777
001712' 777777  775777
001713' 777777  773777
001714' 777777  767777                          EXP     777777767777,777777757777,777777737777
001715' 777777  757777
001716' 777777  737777
001717' 777777  677777                          EXP     777777677777,777777577777,777777377777
```

```
001720' 777777  577777
001721' 777777  377777
001722' 777776  777777                EXP     777776777777,777775777777,777773777777
001723' 777775  777777
001724' 777773  777777
001725' 777767  777777                EXP     777767777777,777757777777,777737777777
001726' 777757  777777
001727' 777737  777777
001730' 777677  777777                EXP     777677777777,777577777777,777377777777
001731' 777577  777777
001732' 777377  777777
001733' 776777  777777                EXP     776777777777,775777777777,773777777777
001734' 775777  777777
001735' 773777  777777
001736' 767777  777777                EXP     767777777777,757777777777,737777777777
001737' 757777  777777
001740' 737777  777777
001741' 677777  777777                EXP     677777777777,577777777777,377777777777
001742' 577777  777777
001743' 377777  777777
001744' 000000  000777                EXP     000000000777,000000777000,000777000000,77700000000
001745' 000000  777000
001746' 000777  000000
001747' 077700  000000
001750' 777777  777000                EXP     777777777000,777777000777,777000777777,00077777777
001751' 777777  000777
001752' 777000  777777
001753' 000077  777777
001754' 000000  777777                EXP     000000777777,000777000777,777000000777
001755' 000777  000777
001756' 777000  000777
001757' 000777  777000                EXP     000777777000,777000777000,777777000000
001760' 777000  777000
001761' 777777  000000
001762' 000777  777777                EXP     000777777777,777000777777,777777000777
001763' 777000  777777
001764' 777777  000777
001765' 777777  777000                EXP     777777777000,777777777777,000000000000
001766' 777777  777777
001767' 000000  000000
001770' 777777  777777                EXP     -1,0
001771' 000000  000000
001772' 000000  000377                EXP     377                     ; table terminator
```

```

                                         ;#*********************************************************************
                                         ;* TEST 24 - Par Gen - Forcing an Error
                                         ;
                                         ; Description:  Set up CSR register so that data sent to the EBus
                                         ;               Module results in the detection of a parity error
                                         ;               which then appears in the CSR register.  Write  a
                                         ;               series  of  data  patterns and check for a parity
                                         ;               error afterwards.
                                         ;
                                         ; Procedure:    EBUS Reset
                                         ;               Write CSR Bit 20 (Generate EBUS PE)
                                         ;               Write data pattern to EBus module
                                         ;               Read CSR register
                                         ;               Did parity error bit set? if no - error
                                         ;                                         if yes - ok
                                         ;
                                         ;               Repeat for each data pattern.
                                         ;
                                         ; Data Patterns: floating one's         777000,,777000
                                         ;               floating zero's         777777,,000000
                                         ;               floating 777's          000777,,777777
                                         ;               floating 000's          777000,,777777
                                         ;               000000,,777777          777777,,000777
                                         ;               000777,,000777          777777,,777000
                                         ;               777000,,000777          777777,,777777
                                         ;               000777,,777000          000000,,000000
                                         ;
                                         ; Failure:      ---
                                         ;#*********************************************************************

                                         ; Test data

001773' 254 00 0 00 002000'              TSTE24: JRST    TG24                            ; go start test
001774' 400000  000024                           EBUS!ZEBUS!24                           ; test mask
001775' 000000  011271'                          0,,[ASCIZ ^Par Gen - Forcing an Error^]
001776' 011260' 011262'                          [EXP E18,MLAST!E19],,[EXP E16,MLAST!E20]
001777' 777777  777777                           -1                                      ; failure test table

                                         ; Start test

002000' 201 00 0 00 000000'              TG24:   MOVEI   Z2                              ; get address of module start
002001' 260 17 0 00 001573*                      GO      TRACE                           ; handle trace output
002002' 400 06 0 00 000000                       SETZ    6,                              ; clear bit table pointer
```

```

                                  ; First, get the data pattern to write

 002003' 400 15 0 00 000000       TA24:  SETZ    ERFLG,                  ; clear error flag
 002004' 201 00 0 06 000001              MOVEI   1(6)                    ; set subtest number to
 002005' 202 00 0 00 001577*             MOVEM   TSTSUB                  ;   pattern number + 1
 002006' 200 01 0 06 001634'             MOVE    1,T23PAT(6)             ; get pattern to write
 002007' 306 01 0 00 000377              CAIN    1,377                   ; end of table?
 002010' 254 00 0 00 002032'             JRST    TX24                    ; yes - end of test

 002011' 202 01 0 00 010361'             MOVEM   1,SAVDAT                ; save data
 002012' 260 17 0 00 001604*             GO      ERESET                  ; do an EBUS Reset
 002013' 201 01 0 00 300000              MOVEI   1,TSTEBF!GENEPE         ; get 'Test EBUF','Gen EBUS PE' bits
 002014' 260 17 0 00 001606*             GO      LDCSR                   ; write to CSR
 002015' 200 01 0 00 010361'             MOVE    1,SAVDAT                ; get data pattern
 002016' 260 17 0 00 001610*             GO      LDEBUF                  ; write to EBUF
 002017' 260 17 0 00 001611*             GO      RDCSR                   ; write to EBUF
 002020' 474 15 0 00 000000              SETO    ERFLG,                  ; error accessing CSR
 002021' 202 01 0 00 010365'             MOVEM   1,SAVCSR                ; save CSR data
 002022' 606 01 0 00 004000              TRNN    1,EBUSPE                ; parity error set?
 002023' 474 15 0 00 000000              SETO    ERFLG,                  ; no - flag error
 002024' 027 00 0 00 002033'             SCOPER  MA24                    ; print error message
 002025' 254 00 0 00 002003'             JRST    TA24                    ; loop on error
 002026' 254 00 0 00 002032'             JRST    TX24                    ; altmode exit
 002027' 326 15 0 00 002032'             JUMPN   ERFLG,TX24              ; error? yes - exit

                                  ; Increment data pattern pointer

 002030' 350 00 0 00 000006              AOS     6                       ; point to next data pattern
 002031' 254 00 0 00 002003'             JRST    TA24                    ; repeat this segment

                                  ; End of test

 002032' 263 17 0 00 000000       TX24:  RTN                             ; return

                                  ; Error messages

 002033' 240000  002035'          MA24:  CALL!TXNOT!MA24PN
 002034' 270000  010231'                 LAST!CALL!TXALL!MPNT2

 002035' 037 00 0 00 011024'      MA24PN: TMSGC  <Wrote '>
 002036' 200 00 0 00 010361'             MOVE    SAVDAT
 002037' 037 13 0 00 000000              PNTHW
 002040' 037 00 0 00 011277'             TMSG    <' to EBUF with 'Gen Ebus PE' set in CSR.>
 002041' 037 00 0 00 011310'             TMSGC   <PE did not occur.>
 002042' 263 17 0 00 000000              RTN
```

```

                                ;#********************************************************************
                                ;* TEST 25 - Read only CSR Bits
                                ;
                                ; Description:  Verify that the CSR bits specified as  read  only
                                ;               are indeed read only.  These  are bits 00-10, 14-
                                ;               23, 29, 32-35.
                                ;
                                ; Procedure:    KL> EBUS Reset
                                ;               KL> Load test ucode
                                ;               UC> Write CSR bits 00-35 all one's
                                ;               UC> Read the CSR register
                                ;               UC> Put results in EBUF
                                ;               KL> Read EBUF
                                ;               KL> CSR bits ok?
                                ;
                                ; Failure:      If the bits are returned incorrectly, and the
                                ;               logic to read and write the EBUF is ok, then bits
                                ;               in the EMUX may be affected by other bits connected
                                ;               to the EMUX which would be due to misconnections or
                                ;               legs of the mixers tied together for some reason.
                                ;#********************************************************************

                                ; Test data

002043' 254 00 0 00 002050'     TSTE25: JRST    TG25                    ; go start test
002044' 400400  000025                  EBUS!NDMP!ZEBUS!25              ; test mask
002045' 002106' 011314'                 T25M,,[ASCIZ ^Read only CSR Bits^]
002046' 011320' 011322'                 [EXP E10,MLAST!E7],,[EXP E1,MLAST!M11]
002047' 777777  777777                  -1                              ; failure test table

                                ; Start test

002050' 201 00 0 00 000000'     TG25:   MOVEI   Z2                      ; get address of module start
002051' 260 17 0 00 002001*             GO      TRACE                   ; handle trace output
002052' 201 01 0 00 002106'             MOVEI   1,T25M                  ; set up microcode address
002053' 260 17 0 00 001520*             GO      TLOAD                   ; load/verify it
002054' 263 17 0 00 000000              RTN                             ; failed - exit test

                                ; 1st segment of test (Segment A) (Start the port)

002055' 400 15 0 00 000000      TA25:   SETZ    ERFLG,                  ; clear error flag
002056' 201 00 0 00 000001              MOVEI   1                       ; set subtest number
002057' 202 00 0 00 002005*             MOVEM   TSTSUB                  ;   to 1
002060' 201 00 0 00 000010              MOVEI   MPRUN                   ; only set MPRUN bit when
002061' 202 00 0 00 001527*             MOVEM   SDATA                   ;      starting the port
002062' 402 00 0 00 001533*             SETZM   SNEXT                   ; set start address to 0
002063' 260 17 0 00 010277'             GO      MSTART                  ; start up port
002064' 254 00 0 00 002055'             JRST    TA25                    ; loop on error
002065' 254 00 0 00 002103'             JRST    TX25                    ; altmode exit
002066' 326 15 0 00 002103'             JUMPN   ERFLG,TX25              ; error? yes - exit
```

```

                                    ; 2nd segment of test (Segment B) (Read EBUF)

002067' 350 00 0 00 002057*                 AOS     TSTSUB                  ; point to next subtest
002070' 200 01 0 00 011324'                 MOVE    1,[160,,3210]           ; set up correct CSR
002071' 202 01 0 00 010367'                 MOVEM   1,SAVCS2                ;   data
002072' 201 01 0 00 200000                  MOVEI   1,TSTEBF                ; set 'DIAG Test EBUF' and
002073' 260 17 0 00 002014*                 GO      LDCSR                   ;   stop the port
002074' 260 17 0 00 001545*                 GO      RDEBUF                  ; read EBUF
002075' 202 01 0 00 010370'                 MOVEM   1,SAVCS3                ; save EBUF data
002076' 312 01 0 00 010367'                 CAME    1,SAVCS2                ; same as expected
002077' 474 15 0 00 000000                  SETO    ERFLG,                  ; no - flag error
002100' 027 00 0 00 002104'                 SCOPER  MB25                    ; print error message
002101' 254 00 0 00 002055'                 JRST    TA25                    ; loop on error
002102' 254 00 0 00 002103'                 JRST    TX25                    ; altmode exit

                                    ; End of test

002103' 263 17 0 00 000000          TX25:   RTN                             ; return

                                    ; Error message - CSR seen by uproc incorrect

002104' 140000  011325'             MB25:   MSG!TXNOT![ASCIZ /Ucode wrote CSR bits all 1's, then read CSR - CSR incorrect/]
002105' 270000  010254'                     LAST!CALL!TXALL!MPNT5

                                    ; Microcode:    First write 1's to EBUF and to CSR

002106' 000000  010000              T25M:   MWORD   <ADDR=0,JMAP,J=1,SD0,AND,B=2,D=2>         ; 0
002107' 742001  000040
002110' 000100  002004                      MWORD   <CONT,SOB,XNOR,B=2,D=1,OENA,SELE,MGC=4> ; 1
002111' 371001  005340
002112' 000200  040040                      MWORD   <CJP,J=4,CENA,CCGC,SELE,MGC=40>         ; 2
002113' 000400  015060
002114' 000300  020000                      MWORD   <JMAP,J=2>                              ; 3
002115' 000000  000040
002116' 000400  000200                      MWORD   <CONT,SOA,A=1,OR,D=1,SELE,MGC=200>      ; 4
002117' 431010  005340

                                    ;       Read CSR into register 2 and write to EBUF and stop

002120' 000500  070040                      MWORD   <CJP,J=7,CENA,CCGC,SELE,MGC=40>         ; 5
002121' 000400  015060
002122' 000600  050000                      MWORD   <JMAP,J=5>                              ; 6
002123' 000000  000040
002124' 000700  000100                      MWORD   <CONT,SD0,B=2,OR,D=2,SELE,MGC=100>      ; 7
002125' 732001  005340
002126' 001000  002004                      MWORD   <CONT,SOA,A=2,OR,D=1,OENA,SELE,MGC=4>   ; 10
002127' 431020  005340
002130' 001100  110000                      MWORD   <JMAP,J=11>                             ; 11
002131' 000000  000040
002132' 777777  777777                      -1
```

```

                                    ;#*******************************************************************
                                    ;* TEST 26 - CSR 01 'DIAG RQST CSR'
                                    ;
                                    ; Description:  Verify that when the microcode requests access to
                                    ;               the  CSR,  the  KL  cannot  read or write the CSR
                                    ;               register, but gets timeout errors instead.
                                    ;
                                    ; Procedure:    KL> EBUS Reset
                                    ;               KL> Load test ucode
                                    ;               UC> Set 'MPRQSTCSR'
                                    ;               KL> Write CSR (to clear port)
                                    ;               KL> Read CSR
                                    ;               KL> Timeout error?
                                    ;
                                    ; Failure:      CSR Bit 01 is 'EBI1 MPRQSTCSRLTCH' and is set
                                    ;               whenever 'EBI2 MPRQSTCSR' is set which gets
                                    ;               decoded from microword bits.
                                    ;#*******************************************************************

                                    ; Test data

002133' 254 00 0 00 002145'         TSTE26: JRST    TG26                            ; go start test
002134' 400400  000026                      EBUS!NDMP!ZEBUS!26                      ; test mask
002135' 002202' 011341'                     T26M,,[ASCIZ ^CSR 01 'DIAG RQST CSR'^]
002136' 011346' 011351'                     [EXP E3,E1,MLAST!E7],,[EXP E9,MLAST!E10]
002137' 000000  002211'                     TSTE27                                  ; failure test table
002140' 000000  002272'                     TSTE30                                  ;   ...
002141' 000000  002347'                     TSTE31
002142' 000000  002427'                     TSTE32
002143' 000000  002501'                     TSTE33
002144' 777777  777777                      -1

                                    ; Start test

002145' 201 00 0 00 000000'         TG26:   MOVEI   Z2                              ; get address of module start
002146' 260 17 0 00 002051*                 GO      TRACE                           ; handle trace output
002147' 201 01 0 00 002202'                 MOVEI   1,T26M                          ; set up microcode address
002150' 260 17 0 00 002053*                 GO      TLOAD                           ; load/verify it
002151' 263 17 0 00 000000                  RTN                                     ; failed - exit test

                                    ; 1st segment of test (Segment A) (Start the port)

002152' 400 15 0 00 000000          TA26:   SETZ    ERFLG,                          ; clear error flag
002153' 201 00 0 00 000001                  MOVEI   1                               ; set subtest number
002154' 202 00 0 00 002067*                 MOVEM   TSTSUB                          ;   to 1
002155' 201 00 0 00 000010                  MOVEI   MPRUN                           ; only set MPRUN bit when
002156' 202 00 0 00 002061*                 MOVEM   SDATA                           ;    starting the port
002157' 201 00 0 00 000001                  MOVEI   1                               ; set start address to 1
002160' 202 00 0 00 002062*                 MOVEM   SNEXT                           ;    for MSTART
002161' 260 17 0 00 010277'                 GO      MSTART                          ; start up port
002162' 254 00 0 00 002152'                 JRST    TA26                            ; loop on error
002163' 254 00 0 00 002177'                 JRST    TX26                            ; altmode exit
002164' 326 15 0 00 002177'                 JUMPN   ERFLG,TX26                      ; error? yes - exit
```

```

                                        ; 2nd segment of test (Segment B) (Write and read CSR)

002165' 350 00 0 00 002154*                     AOS     TSTSUB                          ; point to next subtest
002166' 400 01 0 00 000000                      SETZ    1,                              ; get 0's
002167' 260 17 0 00 002073*                     GO      LDCSR                           ; write to CSR
002170' 260 17 0 00 002017*                     GO      RDCSR                           ; read it
002171' 334 00 0 00 000000                      SKIPA                                   ; error - good
002172' 474 15 0 00 000000                      SETO    ERFLG,                          ; no error - failed
002173' 202 01 0 00 010365'                     MOVEM   1,SAVCSR                        ; save CSR data
002174' 027 00 0 00 002200'                     SCOPER  MB26                            ; print error message
002175' 254 00 0 00 002152'                     JRST    TA26                            ; loop on error
002176' 254 00 0 00 002177'                     JRST    TX26                            ; altmode exit

                                        ; End of test

002177' 263 17 0 00 000000              TX26:   RTN                                     ; return

                                        ; Error message - CSR accessible when ucode should have it hung

002200' 140000  011353'                 MB26:   MSG!TXNOT![ASCIZ /KL could access CSR after ucode set 'MPRQSTCSR'/]
002201' 270000  010231'                         LAST!CALL!TXALL!MPNT2

                                        ; Microcode:    Request CSR

002202' 000000  020040                  T26M:   MWORD   <ADDR=0,CJP,J=2,CENA,CCGC,SELE,MGC=40>  ; 0
002203' 000400  015060
002204' 000100  000000                          MWORD   <JMAP,J=0>                              ; 1
002205' 000000  000040
002206' 000200  020000                          MWORD   <JMAP,J=2>                              ; 2
002207' 000000  000040
002210' 777777  777777                          -1
```

```

                                            ;#*****************************************************************
                                            ;* TEST 27 - CSR 01 'DIAG RQST CSR'
                                            ;
                                            ; Description:  Verify that CSR bit 01 sets  when  the  microcode
                                            ;               requests access to the CSR.
                                            ;
                                            ; Procedure:    KL> EBUS Reset
                                            ;               KL> Load test ucode
                                            ;               UC> Continually request access to CSR and reads it
                                            ;               KL> Read CSR
                                            ;               KL> CSR bit 01 set?
                                            ;               KL> Repeat the above 2 steps until it happens that
                                            ;                   we read CSR register just before the micro-
                                            ;                   processer is granted access to it.
                                            ;
                                            ; Failure:      CSR Bit 01 is 'EBI1 MPRQSTCSRLTCH' and is set
                                            ;               whenever 'EBI2 MPRQSTCSR' is set which gets
                                            ;               decoded from microword bits.
                                            ;#*****************************************************************

                                            ; Test data

002211' 254 00 0 00 002222'                 TSTE27: JRST    TG27                        ; go start test
002212' 400400  000027                              EBUS!NDMP!ZEBUS!27                  ; test mask
002213' 002261' 011341'                             T27M,,[ASCIZ ^CSR 01 'DIAG RQST CSR'^]
002214' 011365' 011351'                             [EXP MLAST!E3],,[EXP E9,MLAST!E10]
002215' 000000  002272'                             TSTE30                              ; failure test table
002216' 000000  002347'                             TSTE31                              ;   ...
002217' 000000  002427'                             TSTE32
002220' 000000  002501'                             TSTE33
002221' 777777  777777                              -1

                                            ; Start test

002222' 201 00 0 00 000000'                 TG27:   MOVEI   Z2                          ; get address of module start
002223' 260 17 0 00 002146*                         GO      TRACE                       ; handle trace output
002224' 201 01 0 00 002261'                         MOVEI   1,T27M                      ; set up microcode address
002225' 260 17 0 00 002150*                         GO      TLOAD                       ; load/verify it
002226' 263 17 0 00 000000                          RTN                                 ; failed - exit test
```

```

                                      ; 1st segment of test (Segment A) (Start the port)

002227' 400 15 0 00 000000            TA27:   SETZ    ERFLG,                          ; clear error flag
002230' 201 00 0 00 000001                    MOVEI   1                               ; set subtest number
002231' 202 00 0 00 002165*                   MOVEM   TSTSUB                          ;   to 1
002232' 201 00 0 00 000010                    MOVEI   MPRUN                           ; only set MPRUN bit when
002233' 202 00 0 00 002156*                   MOVEM   SDATA                           ;    starting the port
002234' 201 00 0 00 000003                    MOVEI   3                               ; set start address to 3
002235' 202 00 0 00 002160*                   MOVEM   SNEXT                           ;    for MSTART
002236' 260 17 0 00 010277'                   GO      MSTART                          ; start up port
002237' 254 00 0 00 002227'                   JRST    TA27                            ; loop on error
002240' 254 00 0 00 002256'                   JRST    TX27                            ; altmode exit
002241' 326 15 0 00 002256'                   JUMPN   ERFLG,TX27                      ; error? yes - exit

                                      ; 2nd segment of test (Segment B) (Look for bit 01 set in CSR)

002242' 350 00 0 00 002231*                   AOS     TSTSUB                          ; point to next subtest
002243' 201 01 0 00 000200                    MOVEI   1,200                           ; check up to 200 times
002244' 260 17 0 00 002170*           TB27:   GO      RDCSR                           ; read it
002245' 400 01 0 00 000000                    SETZ    1,                              ; error - ensure CSR data cleared
002246' 607 01 0 00 200000                    TLNN    1,(CSRRQS)                      ; set?
002247' 367 01 0 00 002244'                   SOJG    1,TB27                          ; no - keep looking for it
002250' 202 01 0 00 010365'                   MOVEM   1,SAVCSR                        ; save CSR data
002251' 607 01 0 00 200000                    TLNN    1,(CSRRQS)                      ; set yet?
002252' 474 15 0 00 000000                    SETO    ERFLG,                          ; no - set error flag
002253' 027 00 0 00 002257'                   SCOPER  MB27                            ; print error message
002254' 254 00 0 00 002227'                   JRST    TA27                            ; loop on error
002255' 254 00 0 00 002256'                   JRST    TX27                            ; altmode exit

                                      ; End of test

002256' 263 17 0 00 000000            TX27:   RTN                                     ; return

                                      ; Error message - CSR bit 01 did not set

002257' 140000  011366'               MB27:   MSG!TXNOT![ASCIZ /Could not set Bit 01 'CSRRQS' in CSR/]
002260' 270000  010231'                       LAST!CALL!TXALL!MPNT2

                                      ; Microcode:      Request/read CSR

002261' 000000  020040                T27M:   MWORD   <ADDR=0,CJP,J=2,CENA,CCGC,SELE,MGC=40>  ; 0
002262' 000400  015060
002263' 000100  000000                        MWORD   <JMAP,J=0>                              ; 1
002264' 000000  000040
002265' 000200  000100                        MWORD   <CONT,SELE,MGC=100>                     ; 2
002266' 000000  005340
002267' 000300  000000                        MWORD   <JZ>                                    ; 3
002270' 000000  000000
002271' 777777  777777                        -1
```

```

                                        ;#********************************************************************
                                        ;* TEST 30 - CSR 01 'DIAG RQST CSR'
                                        ;
                                        ; Description:  Verify that CSR bit 01 clears when the  microcode
                                        ;               reads the CSR register.
                                        ;
                                        ; Procedure:    KL> EBUS Reset
                                        ;               KL> Load test ucode
                                        ;               UC> Set 'MPRQSTCSR'
                                        ;               UC> Set 'MPREADCSR'
                                        ;               KL> Read CSR
                                        ;               KL> CSR bit 01 cleared?
                                        ;
                                        ; Failure:      CSR Bit 01 is 'EBI1 MPRQSTCSRLTCH' and is set
                                        ;               whenever 'EBI2 MPRQSTCSR' is set which gets
                                        ;               decoded from microword bits.
                                        ;#********************************************************************

                                        ; Test data

 002272' 254 00 0 00 002302'            TSTE30: JRST    TG30                    ; go start test
 002273' 400400  000030                         EBUS!NDMP!ZEBUS!30              ; test mask
 002274' 002336' 011341'                        T30M,,[ASCIZ ^CSR 01 'DIAG RQST CSR'^]
 002275' 011365' 011376'                        [EXP MLAST!E3],,[EXP E9,MLAST!E7]
 002276' 000000  002347'                        TSTE31                          ; failure test taole
 002277' 000000  002427'                        TSTE32                          ;   ...
 002300' 000000  002501'                        TSTE33
 002301' 777777  777777                         -1

                                        ; Start test

 002302' 201 00 0 00 000000'            TG30:   MOVEI   Z2                      ; get address of module start
 002303' 260 17 0 00 002223*                    GO      TRACE                   ; handle trace output
 002304' 201 01 0 00 002336'                    MOVEI   1,T30M                  ; set up microcode address
 002305' 260 17 0 00 002225*                    GO      TLOAD                   ; load/verify it
 002306' 263 17 0 00 000000                     RTN                             ; failed - exit test

                                        ; 1st segment of test (Segment A) (Start the port)

 002307' 400 15 0 00 000000             TA30:   SETZ    ERFLG,                  ; clear error flag
 002310' 201 00 0 00 000001                     MOVEI   1                       ; set subtest number
 002311' 202 00 0 00 002242*                    MOVEM   TSTSUB                  ;   to 1
 002312' 201 00 0 00 000010                     MOVEI   MPRUN                   ; only set MPRUN bit when
 002313' 202 00 0 00 002233*                    MOVEM   SDATA                   ;    starting the port
 002314' 201 00 0 00 000001                     MOVEI   1                       ; set start address to 1
 002315' 202 00 0 00 002235*                    MOVEM   SNEXT                   ;    for MSTART
 002316' 260 17 0 00 010277'                    GO      MSTART                  ; start up port
 002317' 254 00 0 00 002307'                    JRST    TA30                    ; loop on error
 002320' 254 00 0 00 002333'                    JRST    TX30                    ; altmode exit
 002321' 326 15 0 00 002333'                    JUMPN   ERFLG,TX30              ; error? yes - exit
```

```

                                         ; 2nd segment of test (Segment B) (Verify that bit 01 is cleared in CSR)

002322' 350 00 0 00 002311*                     AOS     TSTSUB                          ; point to next subtest
002323' 260 17 0 00 002244*                     GO      RDCSR                           ; read it
002324' 474 15 0 00 000000                      SETO    ERFLG,                          ; error
002325' 603 01 0 00 200000                      TLNE    1,(CSRRQS)                      ; set?
002326' 474 15 0 00 000000                      SETO    ERFLG,                          ; yes - set error flag
002327' 202 01 0 00 010365'                     MOVEM   1,SAVCSR                        ; save CSR data
002330' 027 00 0 00 002334'                     SCOPER  MB30                            ; print error message
002331' 254 00 0 00 002307'                     JRST    TA30                            ; loop on error
002332' 254 00 0 00 002333'                     JRST    TX30                            ; altmode exit

                                         ; End of test

002333' 263 17 0 00 000000              TX30:   RTN                                     ; return

                                         ; Error message - CSR bit 01 set

002334' 140000  011400'                 MB30:   MSG!TXNOT![ASCIZ /CSR bit 01 still set after ucode reads CSR/]
002335' 270000  010231'                         LAST!CALL!TXALL!MPNT2

                                         ; Microcode:     Read CSR

002336' 000000  020040                  T30M:   MWORD   <ADDR=0,CJP,J=2,CENA,CCGC,SELE,MGC=40>  ; 0
002337' 000400  015060
002340' 000100  000000                          MWORD   <JMAP,J=0>                              ; 1
002341' 000000  000040
002342' 000200  000100                          MWORD   <CONT,SELE,MGC=100>                     ; 2
002343' 000000  005340
002344' 000300  030000                          MWORD   <JMAP,J=3>                              ; 3
002345' 000000  000040
002346' 777777  777777                          -1
```

```

                                  ;#*******************************************************************
                                  ;* TEST 31 - CSR 01 'DIAG RQST CSR'
                                  ;
                                  ; Description:  Verify that CSR bit 01 clears when the  microcode
                                  ;               writes the CSR register.
                                  ;
                                  ; Procedure:    KL> EBUS Reset
                                  ;               KL> Load test ucode
                                  ;               UC> Set 'MPRQSTCSR'
                                  ;               UC> Set 'MPLOADCSR'
                                  ;               KL> Read CSR
                                  ;               KL> CSR bit 01 cleared?
                                  ;
                                  ; Failure:      CSR Bit 01 is 'EBI1 MPRQSTCSRLTCH' and is set
                                  ;               whenever 'EBI2 MPRQSTCSR' is set which gets
                                  ;               decoded from microword bits.
                                  ;#*******************************************************************

                                  ; Test data

002347' 254 00 0 00 002356'       TSTE31: JRST    TG31                    ; go start test
002350' 400400  000031                    EBUS!NDMP!ZEBUS!31              ; test mask
002351' 002412' 011341'                   T31M,,[ASCIZ ^CSR 01 'DIAG RQST CSR'^]
002352' 011365' 011376'                   [EXP MLAST!E3],,[EXP E9,MLAST!E7]
002353' 000000  002427'                   TSTE32                          ; failure test taole
002354' 000000  002501'                   TSTE33                          ;   ...
002355' 777777  777777                    -1

                                  ; Start test

002356' 201 00 0 00 000000'       TG31:   MOVEI   Z2                      ; get address of module start
002357' 260 17 0 00 002303*               GO      TRACE                   ; handle trace output
002360' 201 01 0 00 002412'               MOVEI   1,T31M                  ; set up microcode address
002361' 260 17 0 00 002305*               GO      TLOAD                   ; load/verify it
002362' 263 17 0 00 000000                RTN                             ; failed - exit test

                                  ; 1st segment of test (Segment A) (Start the port)

002363' 400 15 0 00 000000        TA31:   SETZ    ERFLG,                  ; clear error flag
002364' 201 00 0 00 000001                MOVEI   1                       ; set subtest number
002365' 202 00 0 00 002322*               MOVEM   TSTSUB                  ;   to 1
002366' 201 00 0 00 000010                MOVEI   MPRUN                   ; only set MPRUN bit when
002367' 202 00 0 00 002313*               MOVEM   SDATA                   ;     starting the port
002370' 201 00 0 00 000005                MOVEI   5                       ; set start address to 5
002371' 202 00 0 00 002315*               MOVEM   SNEXT                   ;     for MSTART
002372' 260 17 0 00 010277'               GO      MSTART                  ; start up port
002373' 254 00 0 00 002363'               JRST    TA31                    ; loop on error
002374' 254 00 0 00 002407'               JRST    TX31                    ; altmode exit
002375' 326 15 0 00 002407'               JUMPN   ERFLG,TX31              ; error? yes - exit
```

```

                                  ; 2nd segment of test (Segment B) (Verify that bit 01 is cleared in CSR)

002376' 350 00 0 00 002365*               AOS     TSTSUB                   ; point to next subtest
002377' 260 17 0 00 002323*               GO      RDCSR                    ; read it
002400' 474 15 0 00 000000                SETO    ERFLG,                   ; error
002401' 603 01 0 00 200000                TLNE    1,(CSRRQS)               ; set?
002402' 474 15 0 00 000000                SETO    ERFLG,                   ; yes - set error flag
002403' 202 01 0 00 010365'               MOVEM   1,SAVCSR                 ; save CSR data
002404' 027 00 0 00 002410'               SCOPER  MB31                     ; print error message
002405' 254 00 0 00 002363'               JRST    TA31                     ; loop on error
002406' 254 00 0 00 002407'               JRST    TX31                     ; altmode exit

                                  ; End of test

002407' 263 17 0 00 000000        TX31:   RTN                              ; return

                                  ; Error message - CSR bit 01 set

002410' 140000  011411'           MB31:   MSG!TXNOT![ASCIZ /CSR bit 01 still set after ucode writes CSR/]
002411' 270000  010231'                   LAST!CALL!TXALL!MPNT2

                                  ; Microcode:    Write CSR

002412' 000000  012004            T31M:   MWORD   <ADDR=0,JMAP,J=1,SOA,A=1,AND,D=1,OENA,SELE,MGC=4>
002413' 441010  005040
002414' 000100  030040                    MWORD   <CJP,J=3,CENA,CCGC,SELE,MGC=40>         ; 1
002415' 000400  015060
002416' 000200  010000                    MWORD   <JMAP,J=1>                              ; 2
002417' 000000  000040
002420' 000300  000200                    MWORD   <CONT,SELE,MGC=200>                     ; 3
002421' 000000  005340
002422' 000400  040000                    MWORD   <JMAP,J=4>                              ; 4
002423' 000000  000040
002424' 000500  000000                    MWORD   <JZ>                                    ; 5
002425' 000000  000000
002426' 777777  777777                    -1
```

```

                                ;#********************************************************************
                                ;* TEST 32 - CSR 01 'DIAG RQST CSR'
                                ;
                                ; Description:  Verify that CSR bit 01 clears when  a  reset  via
                                ;               the CSR register is done.
                                ;
                                ; Procedure:    KL> EBUS Reset
                                ;               KL> Load test ucode
                                ;               UC> Set 'MPRQSTCSR'
                                ;               KL> Set CSR bit 18 ('Clear Port')
                                ;               KL> Read CSR
                                ;               KL> CSR bit 01 cleared?
                                ;
                                ; Failure:      ---
                                ;#********************************************************************

                                ; Test data

002427' 254 00 0 00 002435'     TSTE32: JRST    TG32                    ; go start test
002430' 400400  000032                  EBUS!NDMP!ZEBUS!32              ; test mask
002431' 002472' 011341'                 T32M,,[ASCIZ ^CSR 01 'DIAG RQST CSR'^]
002432' 011422' 011376'                 [EXP E3,MLAST!E6],,[EXP E9,MLAST!E7]
002433' 000000  002501'                 TSTE33                          ; failure test table
002434' 777777  777777                  -1                              ;   ...

                                ; Start test

002435' 201 00 0 00 000000'     TG32:   MOVEI   Z2                      ; get address of module start
002436' 260 17 0 00 002357*             GO      TRACE                   ; handle trace output
002437' 201 01 0 00 002472'             MOVEI   1,T32M                  ; set up microcode address
002440' 260 17 0 00 002361*             GO      TLOAD                   ; load/verify it
002441' 263 17 0 00 000000              RTN                             ; failed - exit test

                                ; 1st segment of test (Segment A) (Start the port)

002442' 400 15 0 00 000000      TA32:   SETZ    ERFLG,                  ; clear error flag
002443' 201 00 0 00 000001              MOVEI   1                       ; set subtest number
002444' 202 00 0 00 002376*             MOVEM   TSTSUB                  ;   to 1
002445' 201 00 0 00 000010              MOVEI   MPRUN                   ; only set MPRUN bit when
002446' 202 00 0 00 002367*             MOVEM   SDATA                   ;    starting the port
002447' 201 00 0 00 000001              MOVEI   1                       ; set start address to 1
002450' 202 00 0 00 002371*             MOVEM   SNEXT                   ;    for MSTART
002451' 260 17 0 00 010277'             GO      MSTART                  ; start up port
002452' 254 00 0 00 002442'             JRST    TA32                    ; loop on error
002453' 254 00 0 00 002467'             JRST    TX32                    ; altmode exit
002454' 326 15 0 00 002467'             JUMPN   ERFLG,TX32              ; error? yes - exit
```

```

                                  ; 2nd segment of test (Segment B) (Verify that bit 01 is cleared in CSR)

002455' 350 00 0 00 002444*               AOS     TSTSUB                  ; point to next subtest
002456' 260 17 0 00 001525*               GO      IPACLR                  ; do a 'port clear'
002457' 260 17 0 00 002377*               GO      RDCSR                   ; read it
002460' 474 15 0 00 000000                SETO    ERFLG,                  ; error
002461' 603 01 0 00 200000                TLNE    1,(CSRRQS)              ; set?
002462' 474 15 0 00 000000                SETO    ERFLG,                  ; yes - set error flag
002463' 202 01 0 00 010365'               MOVEM   1,SAVCSR                ; save CSR data
002464' 027 00 0 00 002470'               SCOPER  MB32                    ; print error message
002465' 254 00 0 00 002442'               JRST    TA32                    ; loop on error
002466' 254 00 0 00 002467'               JRST    TX32                    ; altmode exit

                                  ; End of test

002467' 263 17 0 00 000000        TX32:   RTN                             ; return

                                  ; Error message - CSR bit 01 set

002470' 140000  011424'           MB32:   MSG!TXNOT![ASCIZ /CSR bit 01 still set after 'Port Clear'/]
002471' 270000  010231'                   LAST!CALL!TXALL!MPNT2

                                  ; Microcode:     Request CSR

002472' 000000  020040            T32M:   MWORD   <ADDR=0,CJP,J=2,CENA,CCGC,SELE,MGC=40>  ; 0
002473' 000400  015060
002474' 000100  000000                    MWORD   <JMAP,J=0>                              ; 1
002475' 000000  000040
002476' 000200  020000                    MWORD   <JMAP,J=2>                              ; 2
002477' 000000  000040
002500' 777777  777777                    -1
```

```
                                        ;#*****************************************************************
                                        ;* TEST 33 - CSR 01 'DIAG RQST CSR'
                                        ;
                                        ; Description:  Verify that CSR bit 01 clears when a reset via an
                                        ;               EBUS Reset is done.
                                        ;
                                        ; Procedure:    KL> EBUS Reset
                                        ;               KL> Load test ucode
                                        ;               UC> Set 'MPRQSTCSR'
                                        ;               KL> EBUS Reset
                                        ;               KL> Read CSR
                                        ;               KL> CSR bit 01 cleared?
                                        ;
                                        ; Failure:      ---
                                        ;#*****************************************************************

                                        ; Test data

002501' 254 00 0 00 002506'             TSTE33: JRST    TG33                    ; go start test
002502' 400400  000033                          EBUS!NDMP!ZEBUS!33              ; test mask
002503' 002545' 011341'                         T33M,,[ASCIZ ^CSR 01 'DIAG RQST CSR'^]
002504' 011422' 011376'                         [EXP E3,MLAST!E6],,[EXP E9,MLAST!E7]
002505' 777777  777777                          -1                              ; failure test table

                                        ; Start test

002506' 332 00 0 00 030037              TG33:   SKIPE   USER                    ; user mode?
                                                GO      [MOVE   TSTFLG          ; yes - ensure this test does not
                                                         TLO    (TUSER)         ;    get run (but only if not in
                                                         SKIPN  UDEBUG          ;    debug mode)
                                                         MOVEM  TSTFLG
002507' 260 17 0 00 010511'                              RTN]
002510' 201 00 0 00 000000'                     MOVEI   Z2                      ; get address of module start
002511' 260 17 0 00 002436*                     GO      TRACE                   ; handle trace output
002512' 201 01 0 00 002545'                     MOVEI   1,T33M                  ; set up microcode address
002513' 260 17 0 00 002440*                     GO      TLOAD                   ; load/verify it
002514' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; 1st segment of test (Segment A) (Start the port)

002515' 400 15 0 00 000000              TA33:   SETZ    ERFLG,                  ; clear error flag
002516' 201 00 0 00 000001                      MOVEI   1                       ; set subtest number
002517' 202 00 0 00 002455*                     MOVEM   TSTSUB                  ;   to 1
002520' 201 00 0 00 000010                      MOVEI   MPRUN                   ; only set MPRUN bit when
002521' 202 00 0 00 002446*                     MOVEM   SDATA                   ;    starting the port
002522' 201 00 0 00 000001                      MOVEI   1                       ; set start address to 1
002523' 202 00 0 00 002450*                     MOVEM   SNEXT                   ;    for MSTART
002524' 260 17 0 00 010277'                     GO      MSTART                  ; start up port
002525' 254 00 0 00 002515'                     JRST    TA33                    ; loop on error
002526' 254 00 0 00 002542'                     JRST    TX33                    ; altmode exit
002527' 326 15 0 00 002542'                     JUMPN   ERFLG,TX33              ; error? yes - exit
```

```


                                      ; 2nd segment of test (Segment B) (Verify that bit 01 is cleared in CSR)

002530' 350 00 0 00 002517*                   AOS     TSTSUB                      ; point to next subtest
002531' 260 17 0 00 002012*                   GO      ERESET                      ; do an EBUS Reset
002532' 260 17 0 00 002457*                   GO      RDCSR                       ; read it
002533' 474 15 0 00 000000                    SETO    ERFLG,                      ; error
002534' 603 01 0 00 200000                    TLNE    1,(CSRRQS)                  ; set?
002535' 474 15 0 00 000000                    SETO    ERFLG,                      ; yes - set error flag
002536' 202 01 0 00 010365'                   MOVEM   1,SAVCSR                    ; save CSR data
002537' 027 00 0 00 002543'                   SCOPER  MB33                        ; print error message
002540' 254 00 0 00 002515'                   JRST    TA33                        ; loop on error
002541' 254 00 0 00 002542'                   JRST    TX33                        ; altmode exit

                                      ; End of test

002542' 263 17 0 00 000000            TX33:   RTN                                 ; return

                                      ; Error message - CSR bit 01 set

002543' 140000  011434'               MB33:   MSG!TXNOT![ASCIZ /CSR bit 01 still set after 'EBUS Reset'/]
002544' 270000  010231'                       LAST!CALL!TXALL!MPNT2

                                      ; Microcode:     Request CSR

002545' 000000  020040                T33M:   MWORD   <ADDR=0,CJP,J=2,CENA,CCGC,SELE,MGC=40>  ; 0
002546' 000400  015060
002547' 000100  000000                        MWORD   <JMAP,J=0>                              ; 1
002550' 000000  000040
002551' 000200  020000                        MWORD   <JMAP,J=2>                              ; 2
002552' 000000  000040
002553' 777777  777777                        -1
```

```

                                        ;#****************************************************************************
                                        ;* TEST 34 - CSR 02 'DIAG CSR CHNG'
                                        ;
                                        ; Description:  Verify that CSR bit 02 sets when  the  KL  writes
                                        ;               the  CSR  register and clears when the port reads
                                        ;               it.
                                        ;
                                        ; Procedure:    KL> EBUS Reset
                                        ;               KL> Load test ucode and start it at location 0
                                        ;                   (ie. writing CSR register)
                                        ;               UC> Read CSR register (this turns off CSR 02)
                                        ;               KL> Read CSR register and verify CSR 02 is not set
                                        ;               KL> Write the CSR register
                                        ;               UC> See that CSR changed and read it again
                                        ;               KL> Verify CSR 02 is set
                                        ;
                                        ; Failure:      CSR Bit 02 is 'EBI2 CCCSRCHNG' and is set whenever
                                        ;               the KL writes the CSR register via a CONO CSR or
                                        ;               if an MBUS error occurs.
                                        ;#****************************************************************************

                                        ; Test data

 002554' 254 00 0 00 002561'            TSTE34: JRST    TG34                    ; go start test
 002555' 400400  000034                         EBUS!NDMP!ZEBUS!34              ; test mask
 002556' 002633' 011444'                        T34M,,[ASCIZ ^CSR 02 'DIAG CSR CHNG'^]
 002557' 011451' 011452'                        [EXP MLAST!E8],,[EXP E3,MLAST!E9]
 002560' 777777  777777                         -1                              ; failure test table

                                        ; Start test

 002561' 201 00 0 00 000000'            TG34:   MOVEI   Z2                      ; get address of module start
 002562' 260 17 0 00 002511*                    GO      TRACE                   ; handle trace output
 002563' 201 01 0 00 002633'                    MOVEI   1,T34M                  ; set up microcode address
 002564' 260 17 0 00 002513*                    GO      TLOAD                   ; load/verify it
 002565' 263 17 0 00 000000                     RTN                             ; failed - exit test

                                        ; 1st segment of test (Segment A) (Start the port)

 002566' 400 15 0 00 000000             TA34:   SETZ    ERFLG,                  ; clear error flag
 002567' 201 00 0 00 000001                     MOVEI   1                       ; set subtest number
 002570' 202 00 0 00 002530*                    MOVEM   TSTSUB                  ;   to 1
 002571' 201 00 0 00 000010                     MOVEI   MPRUN                   ; only set MPRUN bit when
 002572' 202 00 0 00 002521*                    MOVEM   SDATA                   ;    starting the port
 002573' 201 00 0 00 000001                     MOVEI   1                       ; set start address to 1
 002574' 202 00 0 00 002523*                    MOVEM   SNEXT                   ;    for MSTART
 002575' 260 17 0 00 010277'                    GO      MSTART                  ; start up port
 002576' 254 00 0 00 002566'                    JRST    TA34                    ; loop on error
 002577' 254 00 0 00 002626'                    JRST    TX34                    ; altmode exit
 002600' 326 15 0 00 002626'                    JUMPN   ERFLG,TX34              ; error? yes - exit
```

```

                                        ; 2nd segment of test (Segment B) (Verify that bit 02 is cleared in CSR)

002601' 350 00 0 00 002570*                     AOS     TSTSUB                  ; point to next subtest
002602' 260 17 0 00 002532*                     GO      RDCSR                   ; read it
002603' 474 15 0 00 000000                      SETO    ERFLG,                  ; error
002604' 603 01 0 00 100000                      TLNE    1,(CSRCHN)              ; bit set?
002605' 474 15 0 00 000000                      SETO    ERFLG,                  ; no - set error flag
002606' 202 01 0 00 010365'                     MOVEM   1,SAVCSR                ; save CSR data
002607' 027 00 0 00 002627'                     SCOPER  MA34                    ; print error message
002610' 254 00 0 00 002566'                     JRST    TA34                    ; loop on error
002611' 254 00 0 00 002626'                     JRST    TX34                    ; altmode exit
002612' 326 15 0 00 002626'                     JUMPN   ERFLG,TX34              ; error? yes - exit

                                        ; 3rd segment of test (Segment C) (Verify that bit 02 is set in CSR)

002613' 350 00 0 00 002601*                     AOS     TSTSUB                  ; point to next subtest
002614' 200 01 0 00 010365'                     MOVE    1,SAVCSR                ; get CSR contents
002615' 260 17 0 00 002167*                     GO      LDCSR                   ; write CSR
002616' 260 17 0 00 002602*                     GO      RDCSR                   ; read it
002617' 474 15 0 00 000000                      SETO    ERFLG,                  ; error
002620' 607 01 0 00 100000                      TLNN    1,(CSRCHN)              ; bit set?
002621' 474 15 0 00 000000                      SETO    ERFLG,                  ; no - set error flag
002622' 202 01 0 00 010365'                     MOVEM   1,SAVCSR                ; save CSR data
002623' 027 00 0 00 002631'                     SCOPER  MB34                    ; print error message
002624' 254 00 0 00 002566'                     JRST    TA34                    ; loop on error
002625' 254 00 0 00 002626'                     JRST    TX34                    ; altmode exit

                                        ; End of test

002626' 263 17 0 00 000000              TX34:   RTN                             ; return

                                        ; Error message - CSR bit 02 not set

002627' 140000  011454'                 MA34:   MSG!TXNOT![ASCIZ /CSR bit 02 not initially cleared/]
002630' 270000  010231'                         LAST!CALL!TXALL!MPNT2

002631' 140000  011463'                 MB34:   MSG!TXNOT![ASCIZ /CSR bit 02 not set after KL writes CSR/]
002632' 270000  010231'                         LAST!CALL!TXALL!MPNT2

                                        ; Microcode:    First read CSR to turn off CSR 02

002633' 000000  020040                  T34M:   MWORD   <ADDR=0,CJP,J=2,CENA,CCGC,SELE,MGC=40>  ; 0
002634' 000400  015060
002635' 000100  000000                          MWORD   <JMAP,J=0>                              ; 1
002636' 000000  000040
002637' 000200  000100                          MWORD   <CONT,SELE,MGC=100>                     ; 2
002640' 000000  005340
002641' 000300  030000                          MWORD   <JMAP,J=3>                              ; 3 done
002642' 000000  000040
002643' 777777  777777                          -1
```

```

                                        ;#*********************************************************************
                                        ;* Test 35 - Constant MUX
                                        ;
                                        ; Description:  Verify various data patterns can be used  by  the
                                        ;               2901's via the Cnst MUX correctly.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load microcode
                                        ;
                                        ;               Single step microcode twice doing a JMAP,CONT
                                        ;               Read EBUF, verify data matches the data pattern
                                        ;                 (middle 16 bits are indeterminate)
                                        ;
                                        ;               Repeat for starting addresses 0,2,4,6,10,12,14,16,
                                        ;                 20,22,24,26,30,32,34,36,40,42,44,46,50,52,54,56
                                        ;                 which is data patterns floating 0's, floating 1's,
                                        ;                 0's,1's, 5252.., and 2525..
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

002644' 254 00 0 00 002661'             TSTE35: JRST    TG35                    ; go start test
002645' 410400  000035                          EBUS!MBUS!NDMP!ZEBUS!35         ; test mask
002646' 002775' 011473'                         T35M,,[ASCIZ ^Constant MUX^]
002647' 011476' 011501'                         [EXP E24,E23,MLAST!M11],,[EXP E7,E20,MLAST!M12]
002650' 000000  003136'                         TSTE36                          ; failure test table
002651' 000000000000#                           TSTA1                           ;   ...
002652' 000000000000#                           TSTA2
002653' 000000000000#                           TSTA3
002654' 000000000000#                           TSTA4
002655' 000000000000#                           TSTA5
002656' 000000000000#                           TSTA6
002657' 000000000000#                           TSTA7
002660' 777777  777777                          -1

                                        ; Start test

002661' 201 00 0 00 000000'             TG35:   MOVEI   Z2                      ; get address of module start
002662' 260 17 0 00 002562*                     GO      TRACE                   ; handle trace output
002663' 201 01 0 00 002775'                     MOVEI   1,T35M                  ; set up microcode address
002664' 260 17 0 00 002564*                     GO      TLOAD                   ; load/verify it
002665' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

002666' 400 15 0 00 000000              TL35:   SETZ    ERFLG,                  ; clear error flag
002667' 260 17 0 00 002456*                     GO      IPACLR                  ; clear port
002670' 201 06 0 00 002711'                     MOVEI   6,TS35                  ; get sstep table address
002671' 402 00 0 00 002613*                     SETZM   TSTSUB                  ; initialize subtest number
```

```

                                    ; Loop on single step table entries

002672' 260 17 0 00 000000*         TA35:   GO      BEXEC                           ; execute table entry
002673' 254 00 0 00 002710'                 JRST    TX35                            ; end of sstep table
002674' 254 00 0 00 002672'                 JRST    TA35                            ; keep looping after call
002675' 255 00 0 00 000000                  JFCL                                    ; error - EBUF data incorrect
002676' 200 01 0 00 000000*                 MOVE    1,CEBUF                         ; get correct data
002677' 404 01 0 00 011504'                 AND     1,[777400,,1777]                ; mask out indeterminate bits
002700' 200 02 0 00 000000*                 MOVE    2,AEBUF                         ; get actual data
002701' 404 02 0 00 011504'                 AND     2,[777400,,1777]                ; mask out indeterminate bits
002702' 312 01 0 00 000002                  CAME    1,2                             ; result correct?
002703' 474 15 0 00 000000                  SETO    ERFLG,                          ; no - set error flag

                                    ; Handle error printouts and scope looping

002704' 027 00 0 00 002772'                 SCOPER  MA35                            ; print error message
002705' 254 00 0 00 002666'                 JRST    TL35                            ; loop on error
002706' 254 00 0 00 002710'                 JRST    TX35                            ; altmode exit
002707' 322 15 0 00 002672'                 JUMPE   ERFLG,TA35                      ; do next sstep table entry

                                    ; End of test

002710' 263 17 0 00 000000          TX35:   RTN                                     ; return

                                    ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

002711' 100200  000002              TS35:   ATABLE  (SSSTRT,2,0,2,400000001)
002712' 000400  000001
002713' 100200  020004                      ATABLE  (SSSTRT,2,2,4,1000000002)
002714' 001000  000002
002715' 100200  040006                      ATABLE  (SSSTRT,2,4,6,2000000004)
002716' 002000  000004
002717' 100200  060010                      ATABLE  (SSSTRT,2,6,10,4000000010)
002720' 004000  000010
002721' 100200  100012                      ATABLE  (SSSTRT,2,10,12,10000000020)
002722' 010000  000020
002723' 100200  120014                      ATABLE  (SSSTRT,2,12,14,20000000040)
002724' 020000  000040
002725' 100200  140016                      ATABLE  (SSSTRT,2,14,16,40000000100)
002726' 040000  000100
002727' 100200  160020                      ATABLE  (SSSTRT,2,16,20,100000000200)
002730' 100000  000200
002731' 100200  200022                      ATABLE  (SSSTRT,2,20,22,200000000400)
002732' 200000  000400
002733' 100200  220024                      ATABLE  (SSSTRT,2,22,24,400000001000)
002734' 400000  001000
002735' 100200  240026                      ATABLE  (SSSTRT,2,24,26,777000001776)
002736' 777000  001776
002737' 100200  260030                      ATABLE  (SSSTRT,2,26,30,776400001775)
002740' 776400  001775
002741' 100200  300032                      ATABLE  (SSSTRT,2,30,32,775400001773)
002742' 775400  001773
002743' 100200  320034                      ATABLE  (SSSTRT,2,32,34,773400001767)
002744' 773400  001767
```

```
002745' 100200  340036                  ATABLE  (SSSTRT,2,34,36,767400001757)
002746' 767400  001757
002747' 100200  360040                  ATABLE  (SSSTRT,2,36,40,757400001737)
002750' 757400  001737
002751' 100200  400042                  ATABLE  (SSSTRT,2,40,42,737400001677)
002752' 737400  001677
002753' 100200  420044                  ATABLE  (SSSTRT,2,42,44,677400001577)
002754' 677400  001577
002755' 100200  440046                  ATABLE  (SSSTRT,2,44,46,577400001377)
002756' 577400  001377
002757' 100200  460050                  ATABLE  (SSSTRT,2,46,50,377400000777)
002760' 377400  000777
002761' 100200  500052                  ATABLE  (SSSTRT,2,50,52,000000000000)
002762' 000000  000000
002763' 100200  520054                  ATABLE  (SSSTRT,2,52,54,777400001777)
002764' 777400  001777
002765' 100200  540056                  ATABLE  (SSSTRT,2,54,56,525000001252)
002766' 525000  001252
002767' 100200  560060                  ATABLE  (SSSTRT,2,56,60,252400000525)
002770' 252400  000525
002771' 000000  000000                  ATABLE  (SSLAST)

                                ; Error messages

002772' 140000  011505'         MA35:   MSG!TXNOT![ASCIZ /Constant not gated through CNST MUX correctly/]
002773' 140000  011517'                 MSG!TXNOT![ASCIZ /Result in EBUF (Middle 16 bits indeterminate)/]
002774' 000000000000#                   LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                ; Microcode:

002775' 000000  010001          T35M:   MWORD   <ADDR=0,JMAP,J=1,SDO,OR,D=2,SKCN,MGC=1> ; 0
002776' 732000  240040
002777' 000100  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>       ; 1
003000' 431000  005340
003001' 000200  030002                  MWORD   <JMAP,J=3,SDO,OR,D=2,SKCN,MGC=2>        ; 2
003002' 732000  240040
003003' 000300  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>       ; 3
003004' 431000  005340
003005' 000400  050004                  MWORD   <JMAP,J=5,SDO,OR,D=2,SKCN,MGC=4>        ; 4
003006' 732000  240040
003007' 000500  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>       ; 5
003010' 431000  005340
003011' 000600  070010                  MWORD   <JMAP,J=7,SDO,OR,D=2,SKCN,MGC=10>       ; 6
003012' 732000  240040
003013' 000700  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>       ; 7
003014' 431000  005340
003015' 001000  110020                  MWORD   <JMAP,J=11,SDO,OR,D=2,SKCN,MGC=20>      ; 10
003016' 732000  240040
003017' 001100  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>       ; 11
003020' 431000  005340
003021' 001200  130040                  MWORD   <JMAP,J=13,SDO,OR,D=2,SKCN,MGC=40>      ; 12
003022' 732000  240040
003023' 001300  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>       ; 13
003024' 431000  005340
003025' 001400  150100                  MWORD   <JMAP,J=15,SDO,OR,D=2,SKCN,MGC=100>     ; 14
```

```
003026' 732000  240040
003027' 001500  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>           ; 15
003030' 431000  005340
003031' 001600  170200                  MWORD   <JMAP,J=17,SD0,OR,D=2,SKCN,MGC=200>         ; 16
003032' 732000  240040
003033' 001700  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>           ; 17
003034' 431000  005340
003035' 002000  210400                  MWORD   <JMAP,J=21,SD0,OR,D=2,SKCN,MGC=400>         ; 20
003036' 732000  240040
003037' 002100  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>           ; 21
003040' 431000  005340
003041' 002200  231000                  MWORD   <JMAP,J=23,SD0,OR,D=2,SKCN,MGC=1000>        ; 22
003042' 732000  240040
003043' 002300  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>           ; 23
003044' 431000  005340
003045' 002400  251776                  MWORD   <JMAP,J=25,SD0,OR,D=2,SKCN,MGC=1776>        ; 24
003046' 732000  240040
003047' 002500  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>           ; 25
003050' 431000  005340
003051' 002600  271775                  MWORD   <JMAP,J=27,SD0,OR,D=2,SKCN,MGC=1775>        ; 26
003052' 732000  240040
003053' 002700  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>           ; 27
003054' 431000  005340
003055' 003000  311773                  MWORD   <JMAP,J=31,SD0,OR,D=2,SKCN,MGC=1773>        ; 30
003056' 732000  240040
003057' 003100  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>           ; 31
003060' 431000  005340
003061' 003200  331767                  MWORD   <JMAP,J=33,SD0,OR,D=2,SKCN,MGC=1767>        ; 32
003062' 732000  240040
003063' 003300  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>           ; 33
003064' 431000  005340
003065' 003400  351757                  MWORD   <JMAP,J=35,SD0,OR,D=2,SKCN,MGC=1757>        ; 34
003066' 732000  240040
003067' 003500  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>           ; 35
003070' 431000  005340
003071' 003600  371737                  MWORD   <JMAP,J=37,SD0,OR,D=2,SKCN,MGC=1737>        ; 36
003072' 732000  240040
003073' 003700  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>           ; 37
003074' 431000  005340
003075' 004000  411677                  MWORD   <JMAP,J=41,SD0,OR,D=2,SKCN,MGC=1677>        ; 40
003076' 732000  240040
003077' 004100  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>           ; 41
003100' 431000  005340
003101' 004200  431577                  MWORD   <JMAP,J=43,SD0,OR,D=2,SKCN,MGC=1577>        ; 42
003102' 732000  240040
003103' 004300  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>           ; 43
003104' 431000  005340
003105' 004400  451377                  MWORD   <JMAP,J=45,SD0,OR,D=2,SKCN,MGC=1377>        ; 44
003106' 732000  240040
003107' 004500  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>           ; 45
003110' 431000  005340
003111' 004600  470777                  MWORD   <JMAP,J=47,SD0,OR,D=2,SKCN,MGC=777>         ; 46
003112' 732000  240040
003113' 004700  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>           ; 47
003114' 431000  005340
```

```
  2858  003115' 005000  510000                    MWORD   <JMAP,J=51,SDO,OR,D=2,SKCN,MGC=0>       ; 50
  2859  003116' 732000  240040
  2860  003117' 005100  002004                    MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>       ; 51
  2861  003120' 431000  005340
  2862  003121' 005200  531777                    MWORD   <JMAP,J=53,SDO,OR,D=2,SKCN,MGC=1777>    ; 52
  2863  003122' 732000  240040
  2864  003123' 005300  002004                    MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>       ; 53
  2865  003124' 431000  005340
  2866  003125' 005400  551252                    MWORD   <JMAP,J=55,SDO,OR,D=2,SKCN,MGC=1252>    ; 54
  2867  003126' 732000  240040
  2868  003127' 005500  002004                    MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>       ; 55
  2869  003130' 431000  005340
  2870  003131' 005600  570525                    MWORD   <JMAP,J=57,SDO,OR,D=2,SKCN,MGC=525>     ; 56
  2871  003132' 732000  240040
  2872  003133' 005700  002004                    MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>       ; 57
  2873  003134' 431000  005340
  2874  003135' 777777  777777                    -1
```

```

                                        ;#*********************************************************************
                                        ;* Test 36 - Constant MUX Interference
                                        ;
                                        ; Description:  Verify that the CNST MUX is not selected when the
                                        ;               skip field selection of the constant MUX  is  not
                                        ;               asserted.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load microcode
                                        ;
                                        ;               Single step microcode 4 times doing a JMAP,CONT,
                                        ;                 CONT,CONT
                                        ;               Read EBUF, verify data is zero.
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

003136' 254 00 0 00 003152'             TSTE36: JRST    TG36                    ; go start test
003137' 400400  000036                          EBUS!NDMP!ZEBUS!36              ; test mask
003140' 003201' 011531'                         T36M,,[ASCIZ ^Constant MUX Interference^]
003141' 011537' 011541'                         [EXP E24,MLAST!M11],,[EXP E23,MLAST!M12]
003142' 000000000000#                           TSTA1                           ; failure test table
003143' 000000000000#                           TSTA2                           ;   ...
003144' 000000000000#                           TSTA3
003145' 000000000000#                           TSTA4
003146' 000000000000#                           TSTA5
003147' 000000000000#                           TSTA6
003150' 000000000000#                           TSTA7
003151' 777777  777777                          -1

                                        ; Start test

003152' 201 00 0 00 000000'             TG36:   MOVEI   Z2                      ; get address of module start
003153' 260 17 0 00 002662*                     GO      TRACE                   ; handle trace output
003154' 201 01 0 00 003201'                     MOVEI   1,T36M                  ; set up microcode address
003155' 260 17 0 00 002664*                     GO      TLOAD                   ; load/verify it
003156' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

003157' 400 15 0 00 000000              TL36:   SETZ    ERFLG,                  ; clear error flag
003160' 260 17 0 00 002667*                     GO      IPACLR                  ; clear port
003161' 201 06 0 00 003174'                     MOVEI   6,TS36                  ; get sstep table address
```

```

                                        ; Loop on single step table entries

003162' 260 17 0 00 002672*     TA36:   GO      BEXEC                   ; execute table entry
003163' 254 00 0 00 003173'             JRST    TX36                    ; end of sstep table
003164' 254 00 0 00 003162'             JRST    TA36                    ; keep looping after call
003165' 474 15 0 00 000000              SETO    ERFLG,                  ; error - EBUF data incorrect

                                        ; Handle error printouts and scope looping

003166' 402 00 0 00 002671*             SETZM   TSTSUB                  ; clear subtest number
003167' 027 00 0 00 003177'             SCOPER  MA36                    ; print error message
003170' 254 00 0 00 003157'             JRST    TL36                    ; loop on error
003171' 254 00 0 00 003173'             JRST    TX36                    ; altmode exit
003172' 322 15 0 00 003162'             JUMPE   ERFLG,TA36              ; do next sstep table entry

                                        ; End of test

003173' 263 17 0 00 000000      TX36:   RTN                             ; return

                                        ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

003174' 100400  000004          TS36:   ATABLE  (SSSTRT,4,0,4,0)
003175' 000000  000000
003176' 000000  000000                  ATABLE  (SSLAST)

                                        ; Error messages

003177' 140000  011543'         MA36:   MSG!TXNOT![ASCIZ /CNST MUX seems to interfere with loading of 2901's/]
003200' 000000000000#                   LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                        ; Microcode:

003201' 000000  010000          T36M:   MWORD   <ADDR=0,JMAP,J=1,SDO,AND,D=2>           ; 0
003202' 742000  000040
003203' 000100  002777                  MWORD   <CONT,SOA,OR,D=2,OENA,LDLM,MGC=777>     ; 1
003204' 432000  230340
003205' 000200  000777                  MWORD   <CONT,SDO,OR,D=2,RDLM,MGC=777>          ; 2
003206' 732000  220340
003207' 000300  002004                  MWORD   <CONT,SOA,OR,D=1,SELE,MGC=4,OENA>       ; 3
003210' 431000  005340
003211' 777777  777777                  -1
```

```

                                        ;#*****************************************************************
                                        ;* TEST 37 - CSR Bit 24 - EBUS PE Occurs at Wrong Time
                                        ;
                                        ; Description:  Set up CSR register so that data sent to the EBus
                                        ;               Module results in the detection of a parity error
                                        ;               which  then  appears  in  the CSR register.  Read
                                        ;               EBUF, then read  CSR  and  verify that an EBUS PE
                                        ;               has not occurred.
                                        ;
                                        ; Procedure:    EBUS Reset
                                        ;               Write CSR Bit 19,20,24 (Set up to access EBUF,
                                        ;                 generate EBUS PE, and clear the parity error
                                        ;                 caused by writing the CSR register)
                                        ;               Read EBUF
                                        ;               Read CSR register
                                        ;               Did parity error bit set? if so - error
                                        ;                                         if not - ok
                                        ;
                                        ; Failure:      ---
                                        ;#*****************************************************************

                                        ; Test data

        003212' 254 00 0 00 003217'     TSTE37: JRST    TG37                    ; go start test
        003213' 400000  000037                  EBUS!ZEBUS!37                   ; test mask
        003214' 000000  011556'                 0,,[ASCIZ ^CSR Bit 24 - EBUS PE Occurs at Wrong Time^]
        003215' 011567' 011570'                 [EXP MLAST!E1],,[EXP MLAST!E4]
        003216' 777777  777777                  -1                              ; failure test table

                                        ; Start test

        003217' 201 00 0 00 000000'     TG37:   MOVEI   Z2                      ; get address of module start
        003220' 260 17 0 00 003153*             GO      TRACE                   ; handle trace output

        003221' 400 15 0 00 000000      TA37:   SETZ    ERFLG,                  ; clear error flag
        003222' 260 17 0 00 002531*             GO      ERESET                  ; do an EBUS Reset
        003223' 201 01 0 00 304000              MOVEI   1,TSTEBF!GENEPE!EBUSPE  ; get CSR bits
        003224' 260 17 0 00 002615*             GO      LDCSR                   ; write to CSR
        003225' 260 17 0 00 002074*             GO      RDEBUF                  ; read EBUF
        003226' 260 17 0 00 002616*             GO      RDCSR                   ; read CSR
        003227' 474 15 0 00 000000              SETO    ERFLG,                  ; error accessing CSR
        003230' 202 01 0 00 010365'             MOVEM   1,SAVCSR                ; save CSR data
        003231' 602 01 0 00 004000              TRNE    1,EBUSPE                ; parity error set?
        003232' 474 15 0 00 000000              SETO    ERFLG,                  ; yes - flag error
        003233' 027 00 0 00 003237'             SCOPER  MA37                    ; print error message
        003234' 254 00 0 00 003221'             JRST    TA37                    ; loop on error
        003235' 254 00 0 00 003236'             JRST    TX37                    ; altmode exit
```

```

                                 ; End of test

 003236' 263 17 0 00 000000      TX37:   RTN                                ; return

                                 ; Error messages

 003237' 140000  011571'         MA37:   MSG!TXNOT![ASCIZ /CSR Bit 24 'EBUS PE' set without writing anything over EBUS/]
 003240' 270000  010231'                 LAST!CALL!TXALL!MPNT2
```

```

                                        ;#**********************************************************************
                                        ;* TEST 40 - EBUS Transfer via DATAO's
                                        ;
                                        ; Description:  Transfer data from the KL to the port by using  a
                                        ;               DATAO.  This verifies that IOP function 'examine'
                                        ;               may work, but without using the interrupt system.
                                        ;
                                        ; Procedure:    KL> Port Clear
                                        ;               KL> Start up port
                                        ;               KL> Immediately do a DATAO of data pattern
                                        ;               UC> Wait for 'CCEBUSRQST', then read EBUS data
                                        ;                   into EBUF
                                        ;               KL> Stop the port, read EBUF and verify the data
                                        ;
                                        ; Data Pattern: 0's     525252,,525252
                                        ;               1's     252525,,252525
                                        ;
                                        ; Failure:      ---
                                        ;#**********************************************************************

                                        ; Test data

003241' 254 00 0 00 003247'             TSTE40: JRST    TG40                            ; go start test
003242' 400400  000040                          EBUS!NDMP!ZEBUS!40                      ; test mask
003243' 003336' 011605'                         T40M,,[ASCIZ ^EBUS Transfer via DATAO's^]
003244' 011613' 000000                          [EXP E2,E1,E4,MLAST!E7],,0
003245' 000000  003353'                         TSTE41                                  ; failure test table
003246' 777777  777777                          -1                                      ;   ...

                                        ; Start test

003247' 201 00 0 00 000000'             TG40:   MOVEI   Z2                              ; get address of module start
003250' 260 17 0 00 003220*                     GO      TRACE                           ; handle trace output
003251' 201 01 0 00 003336'                     MOVEI   1,T40M                          ; set up microcode address
003252' 260 17 0 00 003155*                     GO      TLOAD                           ; load/verify it
003253' 263 17 0 00 000000                      RTN                                     ; failed - exit test

                                        ; Initialization/test

003254' 201 06 0 00 003312'                     MOVEI   6,T40DAT                        ; init data pattern pointer

003255' 400 15 0 00 000000              TA40:   SETZ    ERFLG,                          ; clear error flag
003256' 200 00 0 00 000006                      MOVE    6                               ; calculate subtest
003257' 275 00 0 00 003311'                     SUBI    T40DAT-1                        ;   number and save
003260' 202 00 0 00 003166*                     MOVEM   TSTSUB                          ;   it
003261' 260 17 0 00 003160*                     GO      IPACLR                          ; clear port
003262' 400 01 0 00 000000                      SETZ    1,                              ; set up for address 0
003263' 260 17 0 00 000000*                     GO      LDRAR                           ; write the RAR
```

```
003264' 201 01 0 00 000010              MOVEI   1,MPRUN                 ; get 'MPRUN' bit
003265' 260 17 0 00 003224*             GO      LDCSR                   ; write to CSR register
003266' 200 02 0 00 011617'             MOVE    2,[DATAO (6)]           ; build a DATAO
003267' 434 02 0 00 000016              IOR     2,MBCN                  ; include the device code
003270' 336 00 0 00 000000*             SKIPN   UDEBUG                  ; debug mode?
003271' 256 00 0 00 000002              XCT     2                       ; no - do the DATAO
003272' 201 01 0 00 200000              MOVEI   1,TSTEBF                ; set up to read EBUF
003273' 260 17 0 00 003265*             GO      LDCSR                   ; write to CSR register
003274' 260 17 0 00 003225*             GO      RDEBUF                  ; read EBUF
003275' 202 01 0 00 010373'             MOVEM   1,SAVEB2                ; save EBUF data (actual)
003276' 200 00 0 06 000000              MOVE    (6)                     ; get correct data
003277' 202 00 0 00 010372'             MOVEM   SAVEB1                  ; save EBUF data (correct)
003300' 312 00 0 00 000001              CAME    1                       ; data correct?
003301' 474 15 0 00 000000              SETO    ERFLG,                  ; no - flag error

                                ; Handle error printouts and scope looping

003302' 027 00 0 00 003316'             SCOPER  MA40                    ; print error message
003303' 254 00 0 00 003255'             JRST    TA40                    ; loop on error
003304' 254 00 0 00 003311'             JRST    TX40                    ; altmode exit
003305' 326 15 0 00 003311'             JUMPN   ERFLG,TX40              ; error yet? yes - exit test

                                ; Loop on data patterns

003306' 350 00 0 00 000006              AOS     6                       ; point to next data pattern
003307' 307 06 0 00 003315'             CAIG    6,T40DAT+3              ; done yet?
003310' 254 00 0 00 003255'             JRST    TA40                    ; no - loop till done

                                ; End of test

003311' 263 17 0 00 000000      TX40:   RTN                             ; return

                                ; Data Patterns

003312' 000000  000000          T40DAT: 0
003313' 777777  777777                  -1
003314' 525252  525252                  525252525252
003315' 252525  252525                  252525252525
```

```

                                        ; Error messages

003316' 140000  011620'                 MA40:   MSG!TXNOT![ASCIZ /Data from a DATAO not received by port properly/]
003317' 270000  003320'                         LAST!CALL!TXALL!MA40PN          ; print DATAO data

003320' 200 01 0 00 000000*             MA40PN: MOVE    1,SCOSW                 ; get switches
003321' 603 01 0 00 000200                      TLNE    1,TXTINH                ; inhibit text?
003322' 037 00 0 00 011632'                     TMSGC   <DATAO (C):  >          ; yes
003323' 607 01 0 00 000200                      TLNN    1,TXTINH                ; inhibit text?
003324' 037 00 0 00 011635'                     TMSGC   <DATAO (Correct):  >    ; no
003325' 200 00 0 00 010372'                     MOVE    SAVEB1                  ; get DATAO data written
003326' 037 13 0 00 000000                      PNTHW
003327' 603 01 0 00 000200                      TLNE    1,TXTINH                ; inhibit text?
003330' 037 00 0 00 011642'                     TMSGC   <       (A):  >         ; yes
003331' 607 01 0 00 000200                      TLNN    1,TXTINH                ; inhibit text?
003332' 037 00 0 00 011645'                     TMSGC   <         (Actual):  >  ; no
003333' 200 00 0 00 010373'                     MOVE    SAVEB2                  ; get DATAO data
003334' 037 13 0 00 000000                      PNTHW
003335' 263 17 0 00 000000                      RTN

                                        ; Microcode:

003336' 000000  010000                  T40M:   MWORD   <ADDR=0,JMAP,J=1>                       ; 0
003337' 000000  000040
003340' 000100  030000                          MWORD   <CJP,J=3,CENA,CCER>                     ; 1
003341' 000400  100060
003342' 000200  010000                          MWORD   <JMAP,J=1>                              ; 2
003343' 000000  000040
003344' 000300  000010                          MWORD   <CONT,SDO,OR,D=2,SELE,MGC=10>           ; 3
003345' 732000  005340
003346' 000400  002004                          MWORD   <CONT,SOA,OR,D=1,OENA,SELE,MGC=4>       ; 4
003347' 431000  005340
003350' 000500  050000                          MWORD   <JMAP,J=5,D=1>                          ; 5
003351' 001000  000040
003352' 777777  777777                          -1
```

```

                                      ;#***********************************************************************
                                      ;* TEST 41 - EBUS Transfer via DATAI's
                                      ;
                                      ; Description:  Transfer data from the port to the KL by using  a
                                      ;               DATAI.  This verifies that IOP function 'deposit'
                                      ;               may work, but without using the interrupt system.
                                      ;
                                      ; Procedure:    KL> Port Clear
                                      ;               KL> Start up port
                                      ;               KL> Immediately do a DATAI
                                      ;               UC> Wait for 'CCEBUSRQST', then load EBUS data
                                      ;                   with data pattern built and placed in EBUF
                                      ;               KL> Verify result of the DATAI is correct
                                      ;
                                      ; Data Pattern: 0's     525252,,525252
                                      ;               1's     252525,,252525
                                      ;
                                      ; Failure:      ---
                                      ;#***********************************************************************

                                      ; Test data

003353' 254 00 0 00 003360'           TSTE41: JRST    TG41                    ; go start test
003354' 400400  000041                        EBUS!NDMP!ZEBUS!41              ; test mask
003355' 003451' 011652'                       T41M,,[ASCIZ ^EBUS Transfer via DATAI's^]
003356' 011613' 000000                        [EXP E2,E1,E4,MLAST!E7],,0
003357' 777777  777777                        -1                              ; failure test table

                                      ; Start test

003360' 201 00 0 00 000000'           TG41:   MOVEI   Z2                      ; get address of module start
003361' 260 17 0 00 003250*                   GO      TRACE                   ; handle trace output
003362' 201 01 0 00 003451'                   MOVEI   1,T41M                  ; set up microcode address
003363' 260 17 0 00 003252*                   GO      TLOAD                   ; load/verify it
003364' 263 17 0 00 000000                    RTN                             ; failed - exit test

                                      ; Initialization/test

003365' 201 06 0 00 003425'                   MOVEI   6,T41DAT                ; init data pattern pointer

003366' 400 15 0 00 000000            TA41:   SETZ    ERFLG,                  ; clear error flag
003367' 200 00 0 00 000006                    MOVE    6                       ; calculate subtest
003370' 275 00 0 00 003424'                   SUBI    T41DAT-1                ;   number and save
003371' 202 00 0 00 003260*                   MOVEM   TSTSUB                  ;   it
003372' 260 17 0 00 003261*                   GO      IPACLR                  ; clear port
003373' 200 01 0 00 000006                    MOVE    1,6                     ; calculate starting address
003374' 275 01 0 00 003425'                   SUBI    1,T41DAT                ;   0-3
003375' 242 01 0 00 000001                    LSH     1,1                     ; position properly
003376' 260 17 0 00 003263*                   GO      LDRAR                   ; write the RAR
003377' 201 01 0 00 000010                    MOVEI   1,MPRUN                 ; get 'MPRUN' bit
003400' 260 17 0 00 003273*                   GO      LDCSR                   ; write to CSR register
003401' 201 00 0 00 000100                    MOVEI   100                     ; set up delay to allow enough time
003402' 367 00 0 00 003402'                   SOJG    .                       ;   for ucode to prepare for DATAI
```

```
003403' 200 01 0 00 010744'            MOVE    1,[400000,,1]               ; get null pattern
003404' 200 01 0 00 011660'            MOVE    1,[DATAI 2]                 ; build a DATAI
003405' 434 01 0 00 000016             IOR     1,MBCN                      ; include the device code
003406' 336 00 0 00 003270*            SKIPN   UDEBUG                      ; debug mode?
003407' 256 00 0 00 000001             XCT     1                           ; no - do the DATAI
003410' 202 02 0 00 010363'            MOVEM   2,SAVDA2                    ; save DATAI data (actual)
003411' 200 00 0 06 000000             MOVE    (6)                         ; get correct data
003412' 202 00 0 00 010362'            MOVEM   SAVDA1                      ; save DATAI data (correct)
003413' 312 00 0 00 000002             CAME    2                           ; data correct?
003414' 474 15 0 00 000000             SETO    ERFLG,                      ; no - flag error

                                ; Handle error printouts and scope looping

003415' 027 00 0 00 003431'            SCOPER  MA41                        ; print error message
003416' 254 00 0 00 003366'            JRST    TA41                        ; loop on error
003417' 254 00 0 00 003424'            JRST    TX41                        ; altmode exit
003420' 326 15 0 00 003424'            JUMPN   ERFLG,TX41                  ; error yet? yes - exit test

                                ; Loop on data patterns

003421' 350 00 0 00 000006             AOS     6                           ; point to next data pattern
003422' 307 06 0 00 003430'            CAIG    6,T41DAT+3                  ; done yet?
003423' 254 00 0 00 003366'            JRST    TA41                        ; no - loop till done

                                ; End of test

003424' 263 17 0 00 000000      TX41:  RTN                                 ; return

                                ; Data Patterns

003425' 000000  000000          T41DAT: 0
003426' 777777  777777                 -1
003427' 525252  525252                 525252525252
003430' 252525  252525                 252525252525

                                ; Error messages

003431' 140000  011661'         MA41:  MSG!TXNOT![ASCIZ /Data for a DATAI not received by KL properly/]
003432' 270000  003433'                LAST!CALL!TXALL!MA41PN              ; print DATAI data

003433' 200 01 0 00 003320*     MA41PN: MOVE   1,SCOSW                     ; get switches
003434' 603 01 0 00 000200             TLNE    1,TXTINH                    ; inhibit text?
003435' 037 00 0 00 011672'            TMSGC   <DATAI (C):  >              ; yes
003436' 607 01 0 00 000200             TLNN    1,TXTINH                    ; inhibit text?
003437' 037 00 0 00 011675'            TMSGC   <DATAI (Correct):  >        ; no
003440' 200 00 0 00 010362'            MOVE    SAVDA1                      ; get DATAI data written
003441' 037 13 0 00 000000             PNTHW
003442' 603 01 0 00 000200             TLNE    1,TXTINH                    ; inhibit text?
003443' 037 00 0 00 011642'            TMSGC   <        (A):  >            ; yes
003444' 607 01 0 00 000200             TLNN    1,TXTINH                    ; inhibit text?
003445' 037 00 0 00 011645'            TMSGC   <        (Actual):  >       ; no
003446' 200 00 0 00 010363'            MOVE    SAVDA2                      ; get DATAI data
003447' 037 13 0 00 000000             PNTHW
003450' 263 17 0 00 000000             RTN
```

```
3251
3252                                      ; Microcode:
3253
3254  003451' 000001  000000             T41M:  MWORD   <ADDR=0,JMAP,J=100,SOA,AND,B=3,D=2>      ; 0  0's
3255  003452' 442001  400040
3256  003453' 000100  100000                    MWORD   <JMAP,J=10,SOA,AND,B=3,D=2>              ; 1  1's
3257  003454' 442001  400040
3258  003455' 000200  200000                    MWORD   <JMAP,J=20,SOA,AND,B=3,D=2>              ; 2  5252..
3259  003456' 442001  400040
3260  003457' 000300  300000                    MWORD   <JMAP,J=30,SOA,AND,B=3,D=2>              ; 3  2525..
3261  003460' 442001  400040
3262
3263  003461' 001001  000000                    MWORD   <ADDR=10,JMAP,J=100,SOA,XNOR,A=3,B=3,D=2> ; 10
3264  003462' 472031  400040
3265
3266  003463' 002000  210000                    MWORD   <ADDR=20,LDCT,J=21,D=1>                  ; 20
3267  003464' 001000  000300
3268  003465' 002100  000000                    MWORD   <CONT,SOB,B=3,PLUS,D=7,CRY>              ; 21
3269  003466' 307001  400740
3270  003467' 002200  210000                    MWORD   <RPCT,J=21,SOB,B=3,OR,D=7>               ; 22
3271  003470' 337001  400220
3272  003471' 002301  000000                    MWORD   <JMAP,J=100,SOB,B=3,PLUS,CRY,D=7>        ; 23
3273  003472' 307001  400440
3274
3275  003473' 003000  210000                    MWORD   <ADDR=30,LDCT,J=21,D=1>                  ; 30
3276  003474' 001000  000300
3277  003475' 003100  000000                    MWORD   <CONT,SOB,B=3,PLUS,D=7,CRY>              ; 31
3278  003476' 307001  400740
3279  003477' 003200  310000                    MWORD   <RPCT,J=31,SOB,B=3,OR,D=7>               ; 32
3280  003500' 337001  400220
3281  003501' 003300  000000                    MWORD   <CONT,SOB,B=3,PLUS,CRY,D=7>              ; 33
3282  003502' 307001  400740
3283  003503' 003401  000000                    MWORD   <JMAP,J=100,SOB,XNOR,B=3,D=2>            ; 34
3284  003504' 372001  400040
3285
3286                                      ;         Write the data to the EBUS
3287
3288  003505' 010000  002004                    MWORD   <ADDR=100,CONT,SOA,A=3,OR,D=1,OENA,SELE,MGC=4>
3289  003506' 431030  005340
3290  003507' 010101  030000                    MWORD   <CJP,J=103,CENA,CCER>                    ; 101
3291  003510' 000400  100060
3292  003511' 010201  010000                    MWORD   <JMAP,J=101>                             ; 102
3293  003512' 000000  000040
3294  003513' 010300  002020                    MWORD   <CONT,D=1,OENA,SELE,MGC=20>              ; 103
3295  003514' 001000  005340
3296  003515' 010401  040000                    MWORD   <JMAP,J=104,D=1>                         ; 104
3297  003516' 001000  000040
3298  003517' 777777  777777                    -1
```

```

                                        ;#*********************************************************************
                                        ;* TEST 42 - CSR 05 'Rqst Interrupt'
                                        ;
                                        ; Description:  Verify that CSR bit 05 can be cleared by an  EBUS
                                        ;               Reset.
                                        ;
                                        ; Procedure:    KL> EBUS Reset
                                        ;               KL> Load test ucode
                                        ;               UC> Set CSR bit 05
                                        ;               KL> EBUS Reset
                                        ;               KL> Verify CSR 05 is now cleared
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

003520' 254 00 0 00 003527'             TSTE42: JRST    TG42                    ; go start test
003521' 400400  000042                          EBUS!NDMP!ZEBUS!42              ; test mask
003522' 003563' 011702'                         T42M,,[ASCIZ ^CSR 05 'Rqst Interrupt'^]
003523' 011707' 011710'                         [EXP MLAST!E14],,[EXP E9,MLAST!E6]
003524' 000000  003572'                         TSTE43                          ; failure test table
003525' 000000  003643'                         TSTE44                          ;   ...
003526' 777777  777777                          -1

                                        ; Start test

003527' 201 00 0 00 000000'             TG42:   MOVEI   Z2                      ; get address of module start
003530' 260 17 0 00 003361*                     GO      TRACE                   ; handle trace output
003531' 201 01 0 00 003563'                     MOVEI   1,T42M                  ; set up microcode address
003532' 260 17 0 00 003363*                     GO      TLOAD                   ; load/verify it
003533' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; 1st segment of test (Segment A) (Start the port)

003534' 400 15 0 00 000000              TA42:   SETZ    ERFLG,                  ; clear error flag
003535' 201 00 0 00 000001                      MOVEI   1                       ; set subtest number
003536' 202 00 0 00 003371*                     MOVEM   TSTSUB                  ;   to 1
003537' 201 00 0 00 000010                      MOVEI   MPRUN                   ; only set MPRUN bit when
003540' 202 00 0 00 002572*                     MOVEM   SDATA                   ;    starting the port
003541' 402 00 0 00 002574*                     SETZM   SNEXT                   ; set start address to 0
003542' 260 17 0 00 010277'                     GO      MSTART                  ; start up port
003543' 254 00 0 00 003534'                     JRST    TA42                    ; loop on error
003544' 254 00 0 00 003560'                     JRST    TX42                    ; altmode exit
003545' 326 15 0 00 003560'                     JUMPN   ERFLG,TX42              ; error? yes - exit
```

```

                                          ; 2nd segment of test (Segment B) (Verify that bit 05 is cleared in CSR)

  003546' 350 00 0 00 003536*       TB42:   AOS     TSTSUB                  ; point to next subtest
  003547' 260 17 0 00 003222*               GO      ERESET                  ; do and EBUS Reset
  003550' 260 17 0 00 003226*               GO      RDCSR                   ; read it
  003551' 474 15 0 00 000000                SETO    ERFLG,                  ; error
  003552' 603 01 0 00 010000                TLNE    1,(RQINT)               ; bit set?
  003553' 474 15 0 00 000000                SETO    ERFLG,                  ; no - set error flag
  003554' 202 01 0 00 010365'               MOVEM   1,SAVCSR                ; save CSR data
  003555' 027 00 0 00 003561'               SCOPER  MB42                    ; print error message
  003556' 254 00 0 00 003534'               JRST    TA42                    ; loop on error
  003557' 254 00 0 00 003560'               JRST    TX42                    ; altmode exit

                                          ; End of test

  003560' 263 17 0 00 000000        TX42:   RTN                             ; return

                                          ; Error message - CSR bit 05 set

  003561' 140000  011712'           MB42:   MSG!TXNOT![ASCIZ /CSR bit 05 set after EBUS Reset/]
  003562' 270000  010231'                   LAST!CALL!TXALL!MPNT2

                                          ; Microcode:    Cause CSR bit 05 to set

  003563' 000000  010000            T42M:   MWORD   <ADDR=0,JMAP,J=1>                       ; 0
  003564' 000000  000040
  003565' 000100  010000                    MWORD   <JMAP,J=1,BAD>                          ; 1
  003566' 000000  000041
  003567' 000200  020000                    MWORD   <JMAP,J=2>                              ; 2
  003570' 000000  000040
  003571' 777777  777777                    -1
```

```

                                        ;#**********************************************************************
                                        ;* TEST 43 - CSR 05 'Rqst Interrupt'
                                        ;
                                        ; Description:  Verify that CSR bit 05 is set after a CRAM Parity
                                        ;               Error.
                                        ;
                                        ; Procedure:    KL> EBUS Reset
                                        ;               KL> Load test ucode with incorrect parity
                                        ;               KL> Start microcode at location 0
                                        ;               KL> Verify CSR 05 is set
                                        ;
                                        ; Failure:      ---
                                        ;#**********************************************************************

                                        ; Test data

003572' 254 00 0 00 003601'             TSTE43: JRST    TG43                    ; go start test
003573' 400400  000043                          EBUS!NDMP!ZEBUS!43              ; test mask
003574' 003634' 011702'                         T43M,,[ASCIZ ^CSR 05 'Rqst Interrupt'^]
003575' 011721' 011723'                         [EXP E14,MLAST!E15],,[EXP E9,E6,MLAST!M12]
003576' 000000  003643'                         TSTE44                          ; failure test table
003577' 000000  003713'                         TSTE45                          ;   ...
003600' 777777  777777                          -1

                                        ; Start test

003601' 201 00 0 00 000000'             TG43:   MOVEI   Z2                      ; get address of module start
003602' 260 17 0 00 003530*                     GO      TRACE                   ; handle trace output
003603' 201 01 0 00 003634'                     MOVEI   1,T43M                  ; set up microcode address
003604' 260 17 0 00 003532*                     GO      TLOAD                   ; load/verify it
003605' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; 1st segment of test (Segment A) (Start the port)

003606' 400 15 0 00 000000              TA43:   SETZ    ERFLG,                  ; clear error flag
003607' 201 00 0 00 000001                      MOVEI   1                       ; set subtest number
003610' 202 00 0 00 003546*                     MOVEM   TSTSUB                  ;   to 1
003611' 201 00 0 00 000010                      MOVEI   MPRUN                   ; only set MPRUN bit when
003612' 202 00 0 00 003540*                     MOVEM   SDATA                   ;    starting the port
003613' 402 00 0 00 003541*                     SETZM   SNEXT                   ; set start address to 0
003614' 260 17 0 00 010277'                     GO      MSTART                  ; start up port
003615' 254 00 0 00 003606'                     JRST    TA43                    ; loop on error
003616' 254 00 0 00 003631'                     JRST    TX43                    ; altmode exit
003617' 326 15 0 00 003631'                     JUMPN   ERFLG,TX43              ; error? yes - exit
```

```

                                        ; 2nd segment of test (Segment B) (Verify that bit 05 is set in CSR)

003620' 350 00 0 00 003610*     TB43:   AOS     TSTSUB                  ; point to next subtest
003621' 260 17 0 00 003550*             GO      RDCSR                   ; read it
003622' 474 15 0 00 000000              SETO    ERFLG,                  ; error
003623' 607 01 0 00 010000              TLNN    1,(RQINT)               ; bit set?
003624' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
003625' 202 01 0 00 010365'             MOVEM   1,SAVCSR                ; save CSR data
003626' 027 00 0 00 003632'             SCOPER  MB43                    ; print error message
003627' 254 00 0 00 003606'             JRST    TA43                    ; loop on error
003630' 254 00 0 00 003631'             JRST    TX43                    ; altmode exit

                                        ; End of test

003631' 263 17 0 00 000000      TX43:   RTN                             ; return

                                        ; Error message - CSR bit 05 not set

003632' 140000  011726'         MB43:   MSG!TXNOT![ASCIZ /CSR bit 05 not set after CRAM PE/]
003633' 270000  010231'                 LAST!CALL!TXALL!MPNT2

                                        ; Microcode:    Set CRAM PE on second instruction

003634' 000000  010000          T43M:   MWORD   <ADDR=0,JMAP,J=1>                       ; 0
003635' 000000  000040
003636' 000100  010000                  MWORD   <JMAP,J=1,BAD>                          ; 1
003637' 000000  000041
003640' 000200  020000                  MWORD   <JMAP,J=2>                              ; 2
003641' 000000  000040
003642' 777777  777777                  -1
```

```
                                        ;#********************************************************************
                                        ;* TEST 44 - CSR 05 'Rqst Interrupt'
                                        ;
                                        ; Description:  Verify that CSR bit 05 is set after an MBUS Error.
                                        ;
                                        ; Procedure.    KL> EBUS Reset
                                        ;               KL> Load test ucode and start at loc 0
                                        ;               UC> Turn on a whole bunch of MBUS drivers
                                        ;               UC> Turn off a whole bunch of MBUS drivers
                                        ;               KL> Verify CSR 05 is set
                                        ;
                                        ; Failure:      ---
                                        ;#********************************************************************

                                        ; Test data

003643' 254 00 0 00 003655'             TSTE44: JRST    TG44                    ; go start test
003644' 400400  000044                          EBUS!NDMP!ZEBUS!44              ; test mask
003645' 003710' 011702'                         T44M,,[ASCIZ ^CSR 05 'Rqst Interrupt'^]
003646' 011721' 011735'                         [EXP E14,MLAST!E15],,[EXP E9,E6,MLAST!M3]
003647' 000000  003762'                         TSTE46                          ; failure test table
003650' 000000000000#                           TSTU10                          ;   ...
003651' 000000000000#                           TSTU11
003652' 000000000000#                           TSTU12
003653' 000000000000#                           TSTU13
003654' 777777  777777                          -1

                                        ; Start test

003655' 201 00 0 00 000000'             TG44:   MOVEI   Z2                      ; get address of module start
003656' 260 17 0 00 003602*                     GO      TRACE                   ; handle trace output
003657' 201 01 0 00 003710'                     MOVEI   1,T44M                  ; set up microcode address
003660' 260 17 0 00 003604*                     GO      TLOAD                   ; load/verify it
003661' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; 1st segment of test (Segment A) (Start the port)

003662' 400 15 0 00 000000              TA44:   SETZ    ERFLG,                  ; clear error flag
003663' 201 00 0 00 000001                      MOVEI   1                       ; set subtest number
003664' 202 00 0 00 003620*                     MOVEM   TSTSUB                  ;   to 1
003665' 201 00 0 00 000010                      MOVEI   MPRUN                   ; only set MPRUN bit when
003666' 202 00 0 00 003612*                     MOVEM   SDATA                   ;    starting the port
003667' 402 00 0 00 003613*                     SETZM   SNEXT                   ; set start address to 0
003670' 260 17 0 00 010277'                     GO      MSTART                  ; start up port
003671' 254 00 0 00 003662'                     JRST    TA44                    ; loop on error
003672' 254 00 0 00 003705'                     JRST    TX44                    ; altmode exit
003673' 326 15 0 00 003705'                     JUMPN   ERFLG,TX44              ; error? yes - exit
```

```

                                      ; 2nd segment of test (Segment B) (Verify that bit 05 is set in CSR)

003674' 350 00 0 00 003664*     TB44:   AOS     TSTSUB                  ; point to next subtest
003675' 260 17 0 00 003621*             GO      RDCSR                   ; read it
003676' 474 15 0 00 000000              SETO    ERFLG,                  ; error
003677' 607 01 0 00 010000              TLNN    1,(RQINT)               ; bit set?
003700' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
003701' 202 01 0 00 010365'             MOVEM   1,SAVCSR                ; save CSR data
003702' 027 00 0 00 003706'             SCOPER  MB44                    ; print error message
003703' 254 00 0 00 003662'             JRST    TA44                    ; loop on error
003704' 254 00 0 00 003705'             JRST    TX44                    ; altmode exit

                                      ; End of test

003705' 263 17 0 00 000000      TX44:   RTN                             ; return

                                      ; Error message - CSR bit 05 not set

003706' 140000  011740'         MB44:   MSG!TXNOT![ASCIZ /CSR bit 05 not set after MBUS Error/]
003707' 270000  010231'                 LAST!CALL!TXALL!MPNT2

                                      ; Microcode:    Turn on several MBUS drivers to cause MBUS  Error
                                      ;               to set.

003710' 000000  002130          T44M:   MWORD   <ADDR=0,JZ,OENA,RDLM,SELE,MGC=130>
003711' 000000  225000
003712' 777777  777777                  -1
```

```

                                        ;#**********************************************************************
                                        ;* TEST 45 - CSR 06 'CRAM Par Err'
                                        ;
                                        ; Description:  Verify that CSR bit 06 is set after a CRAM parity
                                        ;               error.
                                        ;
                                        ; Procedure:    KL> EBUS Reset
                                        ;               KL> Load test ucode with bad parity
                                        ;               KL> Start microcode at location 0
                                        ;               KL> Read CSR and verify CSR Bit 06 is now set
                                        ;
                                        ; Failure:      This bit is set directly from 'MPR7 CRAMPE'.
                                        ;#**********************************************************************

                                        ; Test data

003713' 254 00 0 00 003720'             TSTE45: JRST    TG45                    ; go start test
003714' 400400  000045                          EBUS!NDMP!ZEBUS!45              ; test mask
003715' 003753' 011750'                         T45M,,[ASCIZ ^CSR 06 'CRAM Par Err'^]
003716' 011503' 011755'                         [EXP MLAST!M12],,[EXP E9,E6,MLAST!E16]
003717' 777777  777777                          -1                              ; failure test table

                                        ; Start test

003720' 201 00 0 00 000000'             TG45:   MOVEI   Z2                      ; get address of module start
003721' 260 17 0 00 003656*                     GO      TRACE                   ; handle trace output
003722' 201 01 0 00 003753'                     MOVEI   1,T45M                  ; set up microcode address
003723' 260 17 0 00 003660*                     GO      TLOAD                   ; load/verify it
003724' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; 1st segment of test (Segment A) (Start the port)

003725' 400 15 0 00 000000              TA45:   SETZ    ERFLG,                  ; clear error flag
003726' 201 00 0 00 000001                      MOVEI   1                       ; set subtest number
003727' 202 00 0 00 003674*                     MOVEM   TSTSUB                  ;   to 1
003730' 201 00 0 00 000010                      MOVEI   MPRUN                   ; only set MPRUN bit when
003731' 202 00 0 00 003666*                     MOVEM   SDATA                   ;    starting the port
003732' 402 00 0 00 003667*                     SETZM   SNEXT                   ; set start address to 0
003733' 260 17 0 00 010277'                     GO      MSTART                  ; start up port
003734' 254 00 0 00 003725'                     JRST    TA45                    ; loop on error
003735' 254 00 0 00 003750'                     JRST    TX45                    ; altmode exit
003736' 326 15 0 00 003750'                     JUMPN   ERFLG,TX45              ; error? yes - exit
```

```

                                          ; 2nd segment of test (Segment B) (Verify that bit 06 is set in CSR)

003737' 350 00 0 00 003727*      TB45:   AOS     TSTSUB                  ; point to next subtest
003740' 260 17 0 00 003675*              GO      RDCSR                   ; read it
003741' 474 15 0 00 000000               SETO    ERFLG,                  ; error
003742' 607 01 0 00 004000               TLNN    1,(CRAMPE)              ; bit set?
003743' 474 15 0 00 000000               SETO    ERFLG,                  ; no - set error flag
003744' 202 01 0 00 010365'              MOVEM   1,SAVCSR                ; save CSR data
003745' 027 00 0 00 003751'              SCOPER  MB45                    ; print error message
003746' 254 00 0 00 003725'              JRST    TA45                    ; loop on error
003747' 254 00 0 00 003750'              JRST    TX45                    ; altmode exit

                                 ; End of test

003750' 263 17 0 00 000000       TX45:   RTN                             ; return

                                 ; Error message - CSR bit 06 not set

003751' 140000  011760'          MB45:   MSG!TXNOT![ASCIZ /CSR bit 06 not set after CRAM PE/]
003752' 270000  010231'                  LAST!CALL!TXALL!MPNT2

                                 ; Microcode:    Set CRAM PE on second instruction

003753' 000000  010000           T45M:   MWORD   <ADDR=0,JMAP,J=1>                       ; 0
003754' 000000  000040
003755' 000100  010000                   MWORD   <JMAP,J=1,BAD>                          ; 1
003756' 000000  000041
003757' 000200  020000                   MWORD   <JMAP,J=2>                              ; 2
003760' 000000  000040
003761' 777777  777777                   -1
```

```

                                    ;#*******************************************************************
                                    ;* TEST 46 - CSR 07 'MBUS Err'
                                    ;
                                    ; Description:  Verify that CSR Bit 07 is set when a  MBUS  error
                                    ;               is detected on the MPROC module.
                                    ;
                                    ; Procedure:    KL> EBUS Reset
                                    ;               KL> Load test ucode
                                    ;               UC> Turn on a whole bunch of MBUS drivers
                                    ;               KL> Verify CSR 07 is set
                                    ;
                                    ; Failure:      This bit is set directly from 'MPR7 CRAMPE'.
                                    ;#*******************************************************************

                                    ; Test data

003762' 254 00 0 00 003773'         TSTE46: JRST    TG46                    ; go start test
003763' 410400  000046                      EBUS!MBUS!NDMP!ZEBUS!46         ; test mask
003764' 004030' 011767'                     T46M,,[ASCIZ ^CSR 07 'MBUS Err'^]
003765' 011737' 011710'                     [EXP MLAST!M3],,[EXP E9,MLAST!E6]
003766' 000000000000#                       TSTU10                          ; failure test table
003767' 000000000000#                       TSTU11                          ;   ...
003770' 000000000000#                       TSTU12
003771' 000000000000#                       TSTU13
003772' 777777  777777                      -1

                                    ; Start test

003773' 201 00 0 00 000000'         TG46:   MOVEI   Z2                      ; get address of module start
003774' 260 17 0 00 003721*                 GO      TRACE                   ; handle trace output
003775' 201 01 0 00 004030'                 MOVEI   1,T46M                  ; set up microcode address
003776' 260 17 0 00 003723*                 GO      TLOAD                   ; load/verify it
003777' 263 17 0 00 000000                  RTN                             ; failed - exit test

                                    ; 1st segment of test (Segment A) (Start the port)

004000' 400 15 0 00 000000          TA46:   SETZ    ERFLG,                  ; clear error flag
004001' 201 00 0 00 000001                  MOVEI   1                       ; set subtest number
004002' 202 00 0 00 003737*                 MOVEM   TSTSUB                  ;   to 1
004003' 201 00 0 00 000010                  MOVEI   MPRUN                   ; only set MPRUN bit when
004004' 202 00 0 00 003731*                 MOVEM   SDATA                   ;    starting the port
004005' 402 00 0 00 003732*                 SETZM   SNEXT                   ; set start address to 0
004006' 260 17 0 00 010277'                 GO      MSTART                  ; start up port
004007' 254 00 0 00 004000'                 JRST    TA46                    ; loop on error
004010' 254 00 0 00 004023'                 JRST    TX46                    ; altmode exit
004011' 326 15 0 00 004023'                 JUMPN   ERFLG,TX46              ; error? yes - exit
```

```

                                ; 2nd segment of test (Segment B) (Verify that bit 07 is set in CSR)

004012' 350 00 0 00 004002*     TB46:   AOS     TSTSUB                  ; point to next subtest
004013' 260 17 0 00 003740*             GO      RDCSR                   ; read it
004014' 474 15 0 00 000000              SETO    ERFLG,                  ; error
004015' 607 01 0 00 002000              TLNN    1,(MBERR)               ; bit set?
004016' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
004017' 202 01 0 00 010365'             MOVEM   1,SAVCSR                ; save CSR data
004020' 027 00 0 00 004026'             SCOPER  MB46                    ; print error message
004021' 254 00 0 00 004000'             JRST    TA46                    ; loop on error
004022' 254 00 0 00 004023'             JRST    TX46                    ; altmode exit

                                ; End of test

004023' 263 17 0 00 000000      TX46:   RTN                             ; return

                                ; Error message - Couldn't start port

004024' 140000  011773'         MA46:   MSG!TXNOT![ASCIZ /Failed to start port properly/]
004025' 270000  010272'                 LAST!CALL!TXALL!MPNT6

                                ; Error message - CSR bit 07 not set

004026' 140000  012001'         MB46:   MSG!TXNOT![ASCIZ /CSR bit 07 not set after MBUS Error/]
004027' 270000  010231'                 LAST!CALL!TXALL!MPNT2

                                ; Microcode:    Turn on several MBUS drivers to cause MBUS  Error
                                ;               to set.

004030' 000000  002130          T46M:   MWORD   <ADDR=0,JZ,OENA,RDLM,SELE,MGC=130>
004031' 000000  225000
004032' 777777  777777                  -1
```

```

                                   ;#*********************************************************************
                                   ;* TEST 47 - CSR 11 'Idle' Bit
                                   ;
                                   ; Description:  Verify that CSR Bit 11 can be read and written by
                                   ;               the  microcode,  and  read  by by the KL10 with a
                                   ;               CONI.
                                   ;
                                   ; Procedure:    KL> EBUS Reset
                                   ;               KL> Load test ucode and start it at loc 0
                                   ;               UC> Read CSR and verify bit 11 zero
                                   ;               UC> Write CSR bit 11, read CSR, verify that it set
                                   ;               KL> Read CSR and verify bit 11 is set
                                   ;               KL> Write CSR to indicate to microcode that this
                                   ;                   has been done by the KL
                                   ;               UC> Zero CSR Bit 11, read CSR, verify that it cleared
                                   ;               KL> Read CSR and verify bit 11 zero
                                   ;               KL> EBUS Reset
                                   ;               KL> Read CSR and verify bit 11 zero
                                   ;
                                   ; Failure:      This bit is written by the microcode via the EBUF
                                   ;               and read by the microcode via the MBUS and read
                                   ;               by the KL as the CSR register.
                                   ;#*********************************************************************

                                   ; Test data

004033' 254 00 0 00 004044'        TSTE47: JRST    TG47                    ; go start test
004034' 400400  000047                     EBUS!NDMP!ZEBUS!47              ; test mask
004035' 004132' 012011'                    T47M,,[ASCIZ ^CSR 11 'Idle' Bit^]
004036' 012015' 011710'                    [EXP MLAST!E12],,[EXP E9,MLAST!E6]
004037' 000000  004233'                    TSTE50                          ; failure test table
004040' 000000  004432'                    TSTE51                          ;   ...
004041' 000000  007402'                    TSTE74
004042' 000000  007744'                    TSTE76
004043' 777777  777777                     -1

                                   ; Start test

004044' 201 00 0 00 000000'        TG47:   MOVEI   Z2                      ; get address of module start
004045' 260 17 0 00 003774*                GO      TRACE                   ; handle trace output
004046' 201 01 0 00 004132'                MOVEI   1,T47M                  ; set up microcode address
004047' 260 17 0 00 003776*                GO      TLOAD                   ; load/verify it
004050' 263 17 0 00 000000                 RTN                             ; failed - exit test
```

```

                                        ; 1st segment of test (Segment A) (Start the port)

 004051' 400 15 0 00 000000         TA47:   SETZ    ERFLG,                          ; clear error flag
 004052' 201 00 0 00 000001                 MOVEI   1                               ; set subtest number
 004053' 202 00 0 00 004012*                MOVEM   TSTSUB                          ;   to 1
 004054' 201 00 0 00 000010                 MOVEI   MPRUN                           ; only set MPRUN bit when
 004055' 202 00 0 00 004004*                MOVEM   SDATA                           ;    starting the port
 004056' 402 00 0 00 004005*                SETZM   SNEXT                           ; set start address to 0
 004057' 260 17 0 00 010277'                GO      MSTART                          ; start up port
 004060' 254 00 0 00 004051'                JRST    TA47                            ; loop on error
 004061' 254 00 0 00 004123'                JRST    TX47                            ; altmode exit
 004062' 326 15 0 00 004123'                JUMPN   ERFLG,TX47                      ; error? yes - exit

                                        ; 2nd segment of test (Segment B) (Verify that bit 11 is set in CSR)

 004063' 350 00 0 00 004053*        TB47:   AOS     TSTSUB                          ; point to next subtest
 004064' 260 17 0 00 004013*                GO      RDCSR                           ; read it
 004065' 474 15 0 00 000000                 SETO    ERFLG,                          ; error
 004066' 202 01 0 00 010365'                MOVEM   1,SAVCSR                        ; save CSR data
 004067' 607 01 0 00 000100                 TLNN    1,(IDLE)                        ; bit set?
 004070' 474 15 0 00 000000                 SETO    ERFLG,                          ; no - set error flag
 004071' 027 00 0 00 004124'                SCOPER  MB47                            ; print error message
 004072' 254 00 0 00 004051'                JRST    TA47                            ; loop on error
 004073' 254 00 0 00 004123'                JRST    TX47                            ; altmode exit
 004074' 326 15 0 00 004123'                JUMPN   ERFLG,TX47                      ; error? yes - exit

                                        ; 3rd segment of test (Segment C) (Verify that bit 11 is cleared in CSR)

 004075' 350 00 0 00 004063*        TC47:   AOS     TSTSUB                          ; point to next subtest
 004076' 550 01 0 00 010365'                HRRZ    1,SAVCSR                        ; save CSR data
 004077' 260 17 0 00 003400*                GO      LDCSR                           ; write CSR register
 004100' 260 17 0 00 004064*                GO      RDCSR                           ; read it
 004101' 474 15 0 00 000000                 SETO    ERFLG,                          ; error
 004102' 603 01 0 00 000100                 TLNE    1,(IDLE)                        ; bit set?
 004103' 474 15 0 00 000000                 SETO    ERFLG,                          ; no - set error flag
 004104' 202 01 0 00 010365'                MOVEM   1,SAVCSR                        ; save CSR data
 004105' 027 00 0 00 004126'                SCOPER  MC47                            ; print error message
 004106' 254 00 0 00 004051'                JRST    TA47                            ; loop on error
 004107' 254 00 0 00 004123'                JRST    TX47                            ; altmode exit
 004110' 326 15 0 00 004123'                JUMPN   ERFLG,TX47                      ; error? yes - exit

                                        ; 4th segment of test (Segment D) (Verify that bit 11 is cleared in CSR)

 004111' 350 00 0 00 004075*        TD47:   AOS     TSTSUB                          ; point to next subtest
 004112' 260 17 0 00 003372*                GO      IPACLR                          ; do a 'port clear'
 004113' 260 17 0 00 004100*                GO      RDCSR                           ; read it
 004114' 474 15 0 00 000000                 SETO    ERFLG,                          ; error
 004115' 603 01 0 00 000100                 TLNE    1,(IDLE)                        ; bit set?
 004116' 474 15 0 00 000000                 SETO    ERFLG,                          ; no - set error flag
 004117' 202 01 0 00 010365'                MOVEM   1,SAVCSR                        ; save CSR data
 004120' 027 00 0 00 004126'                SCOPER  MC47                            ; print error message
 004121' 254 00 0 00 004051'                JRST    TA47                            ; loop on error
 004122' 254 00 0 00 004123'                JRST    TX47                            ; altmode exit
```

```

                                      ; End of test

      004123' 263 17 0 00 000000      TX47:   RTN                                     ; return

                                      ; Error message - CSR bit 11 not set

      004124' 140000  012016'         MB47:   MSG!TXNOT![ASCIZ /CSR bit 11 is not set after ucode writes it/]
      004125' 270000  010231'                 LAST!CALL!TXALL!MPNT2

                                      ; Error message - CSR bit 11 set

      004126' 140000  012027'         MC47:   MSG!TXNOT![ASCIZ /CSR bit 11 set after ucode clears it/]
      004127' 270000  010231'                 LAST!CALL!TXALL!MPNT2

                                      ; Error message - CSR bit 11 set

      004130' 140000  012037'         MD47:   MSG!TXNOT![ASCIZ /CSR bit 11 set after EBUS Reset/]
      004131' 270000  010231'                 LAST!CALL!TXALL!MPNT2

                                      ; Microcode:    First, load a 1 bit mask into Register 2  (bit 11
                                      ;               set)

      004132' 000000  010000          T47M:   MWORD   <ADDR=0,JMAP,J=1,SDO,AND,B=2,D=2>       ; 0
      004133' 742001  000040
      004134' 000100  000000                  MWORD   <CONT,SOB,PLUS,B=2,D=2,CRY>             ; 1
      004135' 302001  000740
      004136' 000200  270000                  MWORD   <LDCT,J=27>                             ; 2
      004137' 000000  000300
      004140' 000300  030000                  MWORD   <RPCT,J=3,SAB,PLUS,A=2,B=2,D=2>         ; 3
      004141' 102021  000220

                                      ;       Read CSR into register 1

      004142' 000400  060040                  MWORD   <CJP,J=6,CENA,CCGC,SELE,MGC=40>         ; 4
      004143' 000400  015060
      004144' 000500  040000                  MWORD   <JMAP,J=4>                              ; 5
      004145' 000000  000040
      004146' 000600  000100                  MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>      ; 6
      004147' 732000  405340

                                      ;       Verify CSR bit 11 is zero

      004150' 000700  110000                  MWORD   <CJP,J=11,SAB,A=2,B=1,AND,D=1,CENA,CCFZ> ; 7
      004151' 141420  420060
      004152' 001001  000000                  MWORD   <JMAP,J=100>                             ; 10
      004153' 000000  000040

                                      ;       Write to CSR (set bit 11)

      004154' 001100  002004                  MWORD   <CONT,SOA,A=2,OR,D=1,OENA,SELE,MGC=4>   ; 11
      004155' 431020  005340
      004156' 001200  140040                  MWORD   <CJP,J=14,CENA,CCGC,SELE,MGC=40>        ; 12
      004157' 000400  015060
      004160' 001300  120000                  MWORD   <JMAP,J=12>                             ; 13
```

```
  3836  004161' 000000  000040
  3837  004162' 001400  000200                    MWORD   <CONT,SOA,A=1,OR,D=1,SELE,MGC=200>      ; 14
  3838  004163' 431010  005340
```

```

                                        ;       Read CSR into register 1

004164' 001500  170040                          MWORD   <CJP,J=17,CENA,CCGC,SELE,MGC=40>        ; 15
004165' 000400  015060
004166' 001600  150000                          MWORD   <JMAP,J=15>                             ; 16
004167' 000000  000040
004170' 001700  000100                          MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>      ; 17
004171' 732000  405340

                                        ;       Verify CSR bit 11 is set

004172' 002001  010000                          MWORD   <CJP,J=101,SAB,A=2,B=1,AND,D=1,CENA,CCFZ> ; 20
004173' 141420  420060

                                        ;       Wait for 'CCCSRCHNG' to come up, indicating the  KL  code
                                        ;       has done its bit.

004174' 002100  230000                          MWORD   <CJP,J=23,CENA,CCCC>                    ; 21
004175' 000400  030060
004176' 002200  210000                          MWORD   <JMAP,J=21>                             ; 22
004177' 000000  000040

                                        ;       Write to CSR (clear bit 11)

004200' 002300  002004                          MWORD   <CONT,SOA,A=1,AND,D=1,OENA,SELE,MGC=4>  ; 23
004201' 441010  005340
004202' 002400  260040                          MWORD   <CJP,J=26,CENA,CCGC,SELE,MGC=40>        ; 24
004203' 000400  015060
004204' 002500  240000                          MWORD   <JMAP,J=24>                             ; 25
004205' 000000  000040
004206' 002600  000200                          MWORD   <CONT,SELE,MGC=200>                     ; 26
004207' 000000  005340

                                        ;       Read CSR into register 1

004210' 002700  310040                          MWORD   <CJP,J=31,CENA,CCGC,SELE,MGC=40>        ; 27
004211' 000400  015060
004212' 003000  270000                          MWORD   <JMAP,J=27>                             ; 30
004213' 000000  000040
004214' 003100  000100                          MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>      ; 31
004215' 732000  405340

                                        ;       Verify CSR bit 11 is clear

004216' 003200  340000                          MWORD   <CJP,J=34,SAB,A=2,B=1,AND,D=1,CENA,CCFZ> ; 32
004217' 141420  420060
004220' 003301  020000                          MWORD   <JMAP,J=102>                             ; 33
004221' 000000  000040

                                        ;       Done - loop forever

004222' 003400  340000                          MWORD   <JMAP,J=34>                             ; 34
004223' 000000  000040

```

```
 3894                                    ;       Error 100 - bit 11 not initially zero
 3895                                    ;       Error 101 - bit 11 failed to set
 3896                                    ;       Error 102 - bit 11 failed to clear
 3897
 3898  004224' 010001  000000                    MWORD   <ADDR=100,JMAP,J=100>                  ; 100
 3899  004225' 000000  000040
 3900  004226' 010101  010000                    MWORD   <ADDR=101,JMAP,J=101>                  ; 101
 3901  004227' 000000  000040
 3902  004230' 010201  020000                    MWORD   <ADDR=102,JMAP,J=102>                  ; 102
 3903  004231' 000000  000040
 3904  004232' 777777  777777                    -1
```

```

                                        ;#*******************************************************************
                                        ;* TEST 50 - CSR 12 'Disable Complete' Bit
                                        ;
                                        ; Description:  Verify that CSR Bit 12 can be read and written by
                                        ;               the  microcode,  and  read  by by the KL10 with a
                                        ;               CONI.
                                        ;
                                        ; Procedure:    KL> EBUS Reset
                                        ;               KL> Load test ucode and start it at loc 0
                                        ;               UC> Read CSR and verify bit 12 zero
                                        ;               UC> Write CSR bit 12, read CSR, verify it set
                                        ;               KL> Read CSR and verify bit 12 is set
                                        ;               KL> Write CSR to indicate to microcode that this
                                        ;                   has been done by the KL
                                        ;               UC> Zero CSR Bit 12, read CSR, verify it cleared
                                        ;               KL> Read CSR and verify bit 12 zero
                                        ;               KL> EBUS Reset
                                        ;               KL> Read CSR and verify bit 12 zero
                                        ;
                                        ; Failure:      This bit is written by the microcode via the EBUF
                                        ;               and read by the microcode via the MBUS and read
                                        ;               by the KL as the CSR register.
                                        ;#*******************************************************************

                                        ; Test data

 004233' 254 00 0 00 004243'            TSTE50: JRST    TG50                    ; go start test
 004234' 400400  000050                         EBUS!NDMP!ZEBUS!50              ; test mask
 004235' 004331' 012046'                        T50M,,[ASCIZ ^CSR 12 'Disable Complete' Bit^]
 004236' 012015' 011710'                        [EXP MLAST!E12],,[EXP E9,MLAST!E6]
 004237' 000000  004432'                        TSTE51                          ; failure test table
 004240' 000000  007402'                        TSTE74                          ;   ...
 004241' 000000  007744'                        TSTE76
 004242' 777777  777777                         -1

                                        ; Start test

 004243' 201 00 0 00 000000'            TG50:   MOVEI   Z2                      ; get address of module start
 004244' 260 17 0 00 004045*                    GO      TRACE                   ; handle trace output
 004245' 201 01 0 00 004331'                    MOVEI   1,T50M                  ; set up microcode address
 004246' 260 17 0 00 004047*                    GO      TLOAD                   ; load/verify it
 004247' 263 17 0 00 000000                     RTN                             ; failed - exit test
```

```

                                      ; 1st segment of test (Segment A) (Start the port)

004250' 400 15 0 00 000000            TA50:   SETZ    ERFLG,                  ; clear error flag
004251' 201 00 0 00 000001                    MOVEI   1                       ; set subtest number
004252' 202 00 0 00 004111*                   MOVEM   TSTSUB                  ;   to 1
004253' 201 00 0 00 000010                    MOVEI   MPRUN                   ; only set MPRUN bit when
004254' 202 00 0 00 004055*                   MOVEM   SDATA                   ;    starting the port
004255' 402 00 0 00 004056*                   SETZM   SNEXT                   ; set start address to 0
004256' 260 17 0 00 010277'                   GO      MSTART                  ; start up port
004257' 254 00 0 00 004250'                   JRST    TA50                    ; loop on error
004260' 254 00 0 00 004322'                   JRST    TX50                    ; altmode exit
004261' 326 15 0 00 004322'                   JUMPN   ERFLG,TX50              ; error? yes - exit

                                      ; 2nd segment of test (Segment B) (Verify that bit 12 is set in CSR)

004262' 350 00 0 00 004252*           TB50:   AOS     TSTSUB                  ; point to next subtest
004263' 260 17 0 00 004113*                   GO      RDCSR                   ; read it
004264' 474 15 0 00 000000                    SETO    ERFLG,                  ; error
004265' 202 01 0 00 010365'                   MOVEM   1,SAVCSR                ; save CSR data
004266' 607 01 0 00 000040                    TLNN    1,(DCOMP)               ; bit set?
004267' 474 15 0 00 000000                    SETO    ERFLG,                  ; no - set error flag
004270' 027 00 0 00 004323'                   SCOPER  MB50                    ; print error message
004271' 254 00 0 00 004250'                   JRST    TA50                    ; loop on error
004272' 254 00 0 00 004322'                   JRST    TX50                    ; altmode exit
004273' 326 15 0 00 004322'                   JUMPN   ERFLG,TX50              ; error? yes - exit

                                      ; 3rd segment of test (Segment C) (Verify that bit 12 is cleared in CSR)

004274' 350 00 0 00 004262*           TC50:   AOS     TSTSUB                  ; point to next subtest
004275' 550 01 0 00 010365'                   HRRZ    1,SAVCSR                ; save CSR data
004276' 260 17 0 00 004077*                   GO      LDCSR                   ; write CSR register
004277' 260 17 0 00 004263*                   GO      RDCSR                   ; read it
004300' 474 15 0 00 000000                    SETO    ERFLG,                  ; error
004301' 603 01 0 00 000040                    TLNE    1,(DCOMP)               ; bit set?
004302' 474 15 0 00 000000                    SETO    ERFLG,                  ; no - set error flag
004303' 202 01 0 00 010365'                   MOVEM   1,SAVCSR                ; save CSR data
004304' 027 00 0 00 004325'                   SCOPER  MC50                    ; print error message
004305' 254 00 0 00 004250'                   JRST    TA50                    ; loop on error
004306' 254 00 0 00 004322'                   JRST    TX50                    ; altmode exit
004307' 326 15 0 00 004322'                   JUMPN   ERFLG,TX50              ; error? yes - exit
```

```

                                          ; 4th segment of test (Segment D) (Verify that bit 12 is cleared in CSR)

004310' 350 00 0 00 004274*               TD50:   AOS     TSTSUB                  ; point to next subtest
004311' 260 17 0 00 004112*                       GO      IPACLR                  ; do a 'port clear'
004312' 260 17 0 00 004277*                       GO      RDCSR                   ; read it
004313' 474 15 0 00 000000                        SETO    ERFLG,                  ; error
004314' 603 01 0 00 000040                        TLNE    1,(DCOMP)               ; bit set?
004315' 474 15 0 00 000000                        SETO    ERFLG,                  ; no - set error flag
004316' 202 01 0 00 010365'                       MOVEM   1,SAVCSR                ; save CSR data
004317' 027 00 0 00 004325'                       SCOPER  MC50                    ; print error message
004320' 254 00 0 00 004250'                       JRST    TA50                    ; loop on error
004321' 254 00 0 00 004322'                       JRST    TX50                    ; altmode exit

                                          ; End of test

004322' 263 17 0 00 000000                TX50:   RTN                             ; return

                                          ; Error message - CSR bit 12 not set

004323' 140000  012054'                   MB50:   MSG!TXNOT![ASCIZ /CSR bit 12 is not set after ucode writes it/]
004324' 270000  010231'                           LAST!CALL!TXALL!MPNT2

                                          ; Error message - CSR bit 12 set

004325' 140000  012065'                   MC50:   MSG!TXNOT![ASCIZ /CSR bit 12 set after ucode clears it/]
004326' 270000  010231'                           LAST!CALL!TXALL!MPNT2

                                          ; Error message - CSR bit 12 set

004327' 140000  012075'                   MD50:   MSG!TXNOT![ASCIZ /CSR bit 12 set after EBUS Reset/]
004330' 270000  010231'                           LAST!CALL!TXALL!MPNT2

                                          ; Microcode:     First, load a 1 bit mask into Register 0  (bit 12
                                          ;                set)

004331' 000000  010000                    T50M:   MWORD   <ADDR=0,JMAP,J=1,SDO,AND,B=2,D=2>       ; 0
004332' 742001  000040
004333' 000100  000000                            MWORD   <CONT,SOB,PLUS,B=2,D=2,CRY>             ; 1
004334' 302001  000740
004335' 000200  260000                            MWORD   <LDCT,J=26>                             ; 2
004336' 000000  000300
004337' 000300  030000                            MWORD   <RPCT,J=3,SAB,PLUS,A=2,B=2,D=2>         ; 3
004340' 102021  000220

                                          ;       Read CSR into register 1

004341' 000400  060040                            MWORD   <CJP,J=6,CENA,CCGC,SELE,MGC=40>         ; 4
004342' 000400  015060
004343' 000500  040000                            MWORD   <JMAP,J=4>                              ; 5
004344' 000000  000040
004345' 000600  000100                            MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>      ; 6
004346' 732000  405340

                                          ;       Verify CSR bit 12 is zero
```

```
 4044
 4045   004347' 000700  110000                      MWORD   <CJP,J=11,SAB,A=2,B=1,AND,D=1,CENA,CCFZ> ; 7
 4046   004350' 141420  420060
 4047   004351' 001001  000000                      MWORD   <JMAP,J=100>                             ; 10
 4048   004352' 000000  000040
 4049
 4050                                       ;       Write to CSR (set bit 12)
 4051
 4052   004353' 001100  002004                      MWORD   <CONT,SOA,A=2,OR,D=1,OENA,SELE,MGC=4>   ; 11
 4053   004354' 431020  005340
 4054   004355' 001200  140040                      MWORD   <CJP,J=14,CENA,CCGC,SELE,MGC=40>        ; 12
 4055   004356' 000400  015060
 4056   004357' 001300  120000                      MWORD   <JMAP,J=12>                             ; 13
 4057   004360' 000000  000040
 4058   004361' 001400  000200                      MWORD   <CONT,SOA,A=1,OR,D=1,SELE,MGC=200>      ; 14
 4059   004362' 431010  005340
 4060
 4061                                       ;       Read CSR into register 1
 4062
 4063   004363' 001500  170040                      MWORD   <CJP,J=17,CENA,CCGC,SELE,MGC=40>        ; 15
 4064   004364' 000400  015060
 4065   004365' 001600  150000                      MWORD   <JMAP,J=15>                             ; 16
 4066   004366' 000000  000040
 4067   004367' 001700  000100                      MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>      ; 17
 4068   004370' 732000  405340
 4069
 4070                                       ;       Verify CSR bit 12 is set
 4071
 4072   004371' 002001  010000                      MWORD   <CJP,J=101,SAB,A=2,B=1,AND,D=1,CENA,CCFZ> ; 20
 4073   004372' 141420  420060
 4074
 4075                                       ;       Wait for 'CCCSRCHNG' to come up, indicating the  KL  code
 4076                                       ;       has done its bit.
 4077
 4078   004373' 002100  230000                      MWORD   <CJP,J=23,CENA,CCCC>                    ; 21
 4079   004374' 000400  030060
 4080   004375' 002200  210000                      MWORD   <JMAP,J=21>                             ; 22
 4081   004376' 000000  000040
 4082
 4083                                       ;       Write to CSR (clear bit 12)
 4084
 4085   004377' 002300  002004                      MWORD   <CONT,SOA,A=1,AND,D=1,OENA,SELE,MGC=4>  ; 23
 4086   004400' 441010  005340
 4087   004401' 002400  260040                      MWORD   <CJP,J=26,CENA,CCGC,SELE,MGC=40>        ; 24
 4088   004402' 000400  015060
 4089   004403' 002500  240000                      MWORD   <JMAP,J=24>                             ; 25
 4090   004404' 000000  000040
 4091   004405' 002600  000200                      MWORD   <CONT,SELE,MGC=200>                     ; 26
 4092   004406' 000000  005340
 4093
 4094                                       ;       Read CSR into register 1
 4095
 4096   004407' 002700  310040                      MWORD   <CJP,J=31,CENA,CCGC,SELE,MGC=40>        ; 27
 4097   004410' 000400  015060
 4098   004411' 003000  270000                      MWORD   <JMAP,J=27>                             ; 30
```

```
004412' 000000  000040
004413' 003100  000100                  MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>      ; 31
004414' 732000  405340


                                ;       Verify CSR bit 12 is clear

004415' 003200  340000                  MWORD   <CJP,J=34,SAB,A=2,B=1,AND,D=1,CENA,CCFZ> ; 32
004416' 141420  420060
004417' 003301  020000                  MWORD   <JMAP,J=102>                             ; 33
004420' 000000  000040

                                ;       Done - loop forever

004421' 003400  340000                  MWORD   <JMAP,J=34>                             ; 34
004422' 000000  000040

                                ;       Error 100 - bit 12 not initially zero
                                ;       Error 101 - bit 12 failed to set
                                ;       Error 102 - bit 12 failed to clear

004423' 010001  000000                  MWORD   <ADDR=100,JMAP,J=100>                   ; 100
004424' 000000  000040
004425' 010101  010000                  MWORD   <ADDR=101,JMAP,J=101>                   ; 101
004426' 000000  000040
004427' 010201  020000                  MWORD   <ADDR=102,JMAP,J=102>                   ; 102
004430' 000000  000040
004431' 777777  777777                  -1
```

```

                                          ;#*******************************************************************
                                          ;* TEST 51 - CSR 13 'Enable Complete' Bit
                                          ;
                                          ; Description:  Verify that CSR Bit 13 can be read and written by
                                          ;               the microcode, and read by the KL10 with a CONI.
                                          ;
                                          ; Procedure:    KL> EBUS Reset
                                          ;               KL> Load test ucode and start it at loc 0
                                          ;               UC> Read CSR and verify bit 13 zero
                                          ;               UC> Write CSR bit 13, read CSR and verify it set
                                          ;               KL> Read CSR and verify bit 13 is set
                                          ;               KL> Write CSR to indicate to microcode that
                                          ;                   this has been done by the KL
                                          ;               UC> Zero CSR Bit 13, read CSR, verify it cleared
                                          ;               KL> Read CSR and verify bit 13 zero
                                          ;               KL> EBUS Reset
                                          ;               KL> Read CSR and verify bit 13 zero
                                          ;
                                          ; Failure:      This bit is written by the microcode via the EBUF
                                          ;               and read by the microcode via the MBUS and read
                                          ;               by the KL as the CSR register.
                                          ;#*******************************************************************

                                          ; Test data

004432' 254 00 0 00 004441'               TSTE51: JRST    TG51                    ; go start test
004433' 400400  000051                            EBUS!NDMP!ZEBUS!51              ; test mask
004434' 004527' 012104'                           T51M,,[ASCIZ ^CSR 13 'Enable Complete' Bit^]
004435' 012015' 011710'                           [EXP MLAST!E12],,[EXP E9,MLAST!E6]
004436' 000000  007402'                           TSTE74                          ; failure test table
004437' 000000  007744'                           TSTE76                          ;   ...
004440' 777777  777777                            -1

                                          ; Start test

004441' 201 00 0 00 000000'               TG51:   MOVEI   Z2                      ; get address of module start
004442' 260 17 0 00 004244*                       GO      TRACE                   ; handle trace output
004443' 201 01 0 00 004527'                       MOVEI   1,T51M                  ; set up microcode address
004444' 260 17 0 00 004246*                       GO      TLOAD                   ; load/verify it
004445' 263 17 0 00 000000                        RTN                             ; failed - exit test

                                          ; 1st segment of test (Segment A) (Start the port)

004446' 400 15 0 00 000000                TA51:   SETZ    ERFLG,                  ; clear error flag
004447' 201 00 0 00 000001                        MOVEI   1                       ; set subtest number
004450' 202 00 0 00 004310*                       MOVEM   TSTSUB                  ;   to 1
004451' 201 00 0 00 000010                        MOVEI   MPRUN                   ; only set MPRUN bit when
004452' 202 00 0 00 004254*                       MOVEM   SDATA                   ;    starting the port
004453' 402 00 0 00 004255*                       SETZM   SNEXT                   ; set start address to 0
004454' 260 17 0 00 010277'                       GO      MSTART                  ; start up port
004455' 254 00 0 00 004446'                       JRST    TA51                    ; loop on error
004456' 254 00 0 00 004520'                       JRST    TX51                    ; altmode exit
004457' 326 15 0 00 004520'                       JUMPN   ERFLG,TX51              ; error? yes - exit
```

```

                                        ; 2nd segment of test (Segment B) (Verify that bit 13 is set in CSR)

004460' 350 00 0 00 004450*             TB51:   AOS     TSTSUB                          ; point to next subtest
004461' 260 17 0 00 004312*                     GO      RDCSR                           ; read it
004462' 474 15 0 00 000000                      SETO    ERFLG,                          ; error
004463' 202 01 0 00 010365'                     MOVEM   1,SAVCSR                        ; save CSR data
004464' 607 01 0 00 000020                      TLNN    1,(ECOMP)                       ; bit set?
004465' 474 15 0 00 000000                      SETO    ERFLG,                          ; no - set error flag
004466' 027 00 0 00 004521'                     SCOPER  MB51                            ; print error message
004467' 254 00 0 00 004446'                     JRST    TA51                            ; loop on error
004470' 254 00 0 00 004520'                     JRST    TX51                            ; altmode exit
004471' 326 15 0 00 004520'                     JUMPN   ERFLG,TX51                      ; error? yes - exit

                                        ; 3rd segment of test (Segment C) (Verify that bit 13 is cleared in CSR)

004472' 350 00 0 00 004460*             TC51:   AOS     TSTSUB                          ; point to next subtest
004473' 550 01 0 00 010365'                     HRRZ    1,SAVCSR                        ; save CSR data
004474' 260 17 0 00 004276*                     GO      LDCSR                           ; write CSR register
004475' 260 17 0 00 004461*                     GO      RDCSR                           ; read it
004476' 474 15 0 00 000000                      SETO    ERFLG,                          ; error
004477' 603 01 0 00 000020                      TLNE    1,(ECOMP)                       ; bit set?
004500' 474 15 0 00 000000                      SETO    ERFLG,                          ; no - set error flag
004501' 202 01 0 00 010365'                     MOVEM   1,SAVCSR                        ; save CSR data
004502' 027 00 0 00 004523'                     SCOPER  MC51                            ; print error message
004503' 254 00 0 00 004446'                     JRST    TA51                            ; loop on error
004504' 254 00 0 00 004520'                     JRST    TX51                            ; altmode exit
004505' 326 15 0 00 004520'                     JUMPN   ERFLG,TX51                      ; error? yes - exit

                                        ; 4th segment of test (Segment D) (Verify that bit 13 is cleared in CSR)

004506' 350 00 0 00 004472*             TD51:   AOS     TSTSUB                          ; point to next subtest
004507' 260 17 0 00 004311*                     GO      IPACLR                          ; do a 'port clear'
004510' 260 17 0 00 004475*                     GO      RDCSR                           ; read it
004511' 474 15 0 00 000000                      SETO    ERFLG,                          ; error
004512' 603 01 0 00 000020                      TLNE    1,(ECOMP)                       ; bit set?
004513' 474 15 0 00 000000                      SETO    ERFLG,                          ; no - set error flag
004514' 202 01 0 00 010365'                     MOVEM   1,SAVCSR                        ; save CSR data
004515' 027 00 0 00 004523'                     SCOPER  MC51                            ; print error message
004516' 254 00 0 00 004446'                     JRST    TA51                            ; loop on error
004517' 254 00 0 00 004520'                     JRST    TX51                            ; altmode exit

                                        ; End of test

004520' 263 17 0 00 000000              TX51:   RTN                                     ; return

                                        ; Error message - CSR bit 13 not set

004521' 140000  012112'                 MB51:   MSG!TXNOT![ASCIZ /CSR bit 13 is not set after ucode writes it/]
004522' 270000  010231'                         LAST!CALL!TXALL!MPNT2
```

```

                                        ; Error message - CSR bit 13 set

004523' 140000  012123'         MC51:   MSG!TXNOT![ASCIZ /CSR bit 13 set after ucode clears it/]
004524' 270000  010231'                 LAST!CALL!TXALL!MPNT2

                                        ; Error message - CSR bit 13 set

004525' 140000  012133'         MD51:   MSG!TXNOT![ASCIZ /CSR bit 13 set after EBUS Reset/]
004526' 270000  010231'                 LAST!CALL!TXALL!MPNT2

                                        ; Microcode:    First, load a 1 bit mask into Register 0  (bit 13
                                        ;               set)

004527' 000000  010000          T51M:   MWORD   <ADDR=0,JMAP,J=1,SDO,AND,B=2,D=2>       ; 0
004530' 742001  000040
004531' 000100  000000                  MWORD   <CONT,SOB,PLUS,B=2,D=2,CRY>             ; 1
004532' 302001  000740
004533' 000200  250000                  MWORD   <LDCT,J=25>                             ; 2
004534' 000000  000300
004535' 000300  030000                  MWORD   <RPCT,J=3,SAB,PLUS,A=2,B=2,D=2>         ; 3
004536' 102021  000220

                                        ;       Read CSR into register 1

004537' 000400  060040                  MWORD   <CJP,J=6,CENA,CCGC,SELE,MGC=40>         ; 4
004540' 000400  015060
004541' 000500  040000                  MWORD   <JMAP,J=4>                              ; 5
004542' 000000  000040
004543' 000600  000100                  MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>      ; 6
004544' 752000  405340

                                        ;       Verify CSR bit 13 is zero

004545' 000700  110000                  MWORD   <CJP,J=11,SAB,A=2,B=1,AND,D=1,CENA,CCFZ> ; 7
004546' 141420  420060
004547' 001001  000000                  MWORD   <JMAP,J=100>                             ; 10
004550' 000000  000040

                                        ;       Write to CSR (set bit 13)

004551' 001100  002004                  MWORD   <CONT,SOA,A=2,OR,D=1,OENA,SELE,MGC=4>   ; 11
004552' 431020  005340
004553' 001200  140040                  MWORD   <CJP,J=14,CENA,CCGC,SELE,MGC=40>        ; 12
004554' 000400  015060
004555' 001300  120000                  MWORD   <JMAP,J=12>                             ; 13
004556' 000000  000040
004557' 001400  000200                  MWORD   <CONT,SOA,A=1,OR,D=1,SELE,MGC=200>      ; 14
004560' 431010  005340

                                        ;       Read CSR into register 1

004561' 001500  170040                  MWORD   <CJP,J=17,CENA,CCGC,SELE,MGC=40>        ; 15
004562' 000400  015060
004563' 001600  150000                  MWORD   <JMAP,J=15>                             ; 16
```

```
004564' 000000  000040
004565' 001700  000100                          MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>      ; 17
004566' 732000  405340

                                        ;       Verify CSR bit 13 is set

004567' 002001  010000                          MWORD   <CJP,J=101,SAB,A=2,B=1,AND,D=1,CENA,CCFZ> ; 20
004570' 141420  420060

                                        ;       Wait for 'CCCSRCHNG' to come up, indicating the  KL  code
                                        ;       has done its bit.

004571' 002100  230000                          MWORD   <CJP,J=23,CENA,CCCC>                    ; 21
004572' 000400  030060
004573' 002200  210000                          MWORD   <JMAP,J=21>                             ; 22
004574' 000000  000040

                                        ;       Write to CSR (clear bit 13)

004575' 002300  002004                          MWORD   <CONT,SOA,A=1,AND,D=1,OENA,SELE,MGC=4>  ; 23
004576' 441010  005340
004577' 002400  260040                          MWORD   <CJP,J=26,CENA,CCGC,SELE,MGC=40>        ; 24
004600' 000400  015060
004601' 002500  240000                          MWORD   <JMAP,J=24>                             ; 25
004602' 000000  000040
004603' 002600  000200                          MWORD   <CONT,SELE,MGC=200>                     ; 26
004604' 000000  005340

                                        ;       Read CSR into register 1

004605' 002700  310040                          MWORD   <CJP,J=31,CENA,CCGC,SELE,MGC=40>        ; 27
004606' 000400  015060
004607' 003000  270000                          MWORD   <JMAP,J=27>                             ; 30
004610' 000000  000040
004611' 003100  000100                          MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>      ; 31
004612' 732000  405340

                                        ;       Verify CSR bit 13 is clear

004613' 003200  340000                          MWORD   <CJP,J=34,SAB,A=2,B=1,AND,D=1,CENA,CCFZ> ; 32
004614' 141420  420060
004615' 003301  020000                          MWORD   <JMAP,J=102>                             ; 33
004616' 000000  000040

                                        ;       Done - loop forever

004617' 003400  340000                          MWORD   <JMAP,J=34>                             ; 34
004620' 000000  000040

                                        ;       Error 100 - bit 13 not initially zero
                                        ;       Error 101 - bit 13 failed to set
                                        ;       Error 102 - bit 13 failed to clear

004621' 010001  000000                          MWORD   <ADDR=100,JMAP,J=100>                   ; 100
004622' 000000  000040
```

```
 4340  004623' 010101  010000                 MWORD   <ADDR=101,JMAP,J=101>                   ; 101
 4341  004624' 000000  000040
 4342  004625' 010201  020000                 MWORD   <ADDR=102,JMAP,J=102>                   ; 102
 4343  004626' 000000  000040
 4344  004627' 777777  777777                 -1
```

```

                                        ;#************************************************************************
                                        ;* TEST 52 - x
                                        ;#************************************************************************

                                        ; Test data

004630' 263 17 0 00 000000              TSTE52: RTN                             ; exit
004631' 000000  000052                          ZEBUS!52                        ; test mask
004632' 000000  012142'                         0,,[ASCIZ ^x^]
004633' 000000  000000                          0,,0
004634' 777777  777777                          -1
```

```
                                        ;#*************************************************************
                                        ;* TEST 53 - MPROC Run Bit
                                        ;
                                        ; Description:  This test verifies that once MPROC Run is set  it
                                        ;               stays set.
                                        ;
                                        ; Procedure:    KL> EBUS Reset
                                        ;               KL> Load test ucode (set to start at location 0)
                                        ;               KL> Write CSR to set MPROC Run
                                        ;               KL> Read CSR and verify that MPROC Run is still
                                        ;                   set?  (It should be)
                                        ;
                                        ; Failure:      Once set and if Single Step is not set, MPROC Run
                                        ;               stays set until either cleared by re-writing the
                                        ;               CSR Register or upon occurence of an error.
                                        ;#*************************************************************

                                        ; Test data

004635' 254 00 0 00 004646'             TSTE53: JRST    TG53                    ; go start test
004636' 400400  000053                          EBUS!NDMP!ZEBUS!53              ; test mask
004637' 004701' 012143'                         T53M,,[ASCIZ ^MPROC Run Bit^]
004640' 010652' 012146'                         [EXP MLAST!E17],,[EXP E6,E1,E12,E14,MLAST!M1]
004641' 000000000000#                           TSTU6                           ; failure test table
004642' 000000000000#                           TSTU7                           ;   ...
004643' 000000000000#                           TSTU41
004644' 000000  003713'                         TSTE45
004645' 777777  777777                          -1

                                        ; Start test

004646' 201 00 0 00 000000'             TG53:   MOVEI   Z2                      ; get address of module start
004647' 260 17 0 00 004442*                     GO      TRACE                   ; handle trace output
004650' 201 01 0 00 004701'                     MOVEI   1,T53M                  ; set up microcode address
004651' 260 17 0 00 004444*                     GO      TLOAD                   ; load/verify it
004652' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; 1st segment of test (Segment A) (Start the port)

004653' 400 15 0 00 000000              TA53:   SETZ    ERFLG,                  ; clear error flag
004654' 201 00 0 00 000001                      MOVEI   1                       ; set subtest number
004655' 202 00 0 00 004506*                     MOVEM   TSTSUB                  ;   to 1
004656' 201 00 0 00 000010                      MOVEI   MPRUN                   ; only set MPRUN bit when
004657' 202 00 0 00 004452*                     MOVEM   SDATA                   ;    starting the port
004660' 402 00 0 00 004453*                     SETZM   SNEXT                   ; set start address to 0
004661' 260 17 0 00 010277'                     GO      MSTART                  ; start up port
004662' 254 00 0 00 004653'                     JRST    TA53                    ; loop on error
004663' 254 00 0 00 004676'                     JRST    TX53                    ; altmode exit
004664' 326 15 0 00 004676'                     JUMPN   ERFLG,TX53              ; error? yes - exit
```

```

                                          ; 2nd segment of test (Segment B) (Verify MPRun bit)

004665' 350 00 0 00 004655*        TB53:    AOS     TSTSUB                        ; point to next subtest
004666' 260 17 0 00 004510*                 GO      RDCSR                         ; read CSR
004667' 474 15 0 00 000000                  SETO    ERFLG,                        ; error
004670' 202 01 0 00 010365'                 MOVEM   1,SAVCSR                      ; save data
004671' 606 01 0 00 000010                  TRNN    1,MPRUN                       ; 'MPRun' set?
004672' 474 15 0 00 000000                  SETO    ERFLG,                        ; no - flag error
004673' 027 00 0 00 004677'                 SCOPER  MB53                          ; print error message
004674' 254 00 0 00 004653'                 JRST    TA53                          ; loop on error
004675' 254 00 0 00 004676'                 JRST    TX53                          ; altmode exit

                                   ; End of test

004676' 263 17 0 00 000000         TX53:    RTN                                   ; return

                                   ; Error message - MPRun cleared

004677' 140000  012153'            MB53:    MSG!TXNOT![ASCIZ /CSR Bit 32 'MPRun' did not stay set/]
004700' 270000  010231'                     LAST!CALL!TXALL!MPNT2

                                   ; Microcode:    Infinite loop.

004701' 000000  000000             T53M:    MWORD   <ADDR=0,JZ>
004702' 000000  000000
004703' 777777  777777                      -1
```

```
                                        ;#*****************************************************************
                                        ;* TEST 54 - Basic KMUX Parity Test
                                        ;
                                        ; Description:  Do a CONI and see if a page fail occurred.
                                        ;
                                        ; Procedure:    EBUS Reset (CSR should have odd parity)
                                        ;               Read CSR
                                        ;               Page fail (due to AR/ARx parity error?
                                        ;                 if yes - error
                                        ;                 if no - ok
                                        ;               Write 0's to CSR (should now have even parity)
                                        ;               Read CSR
                                        ;               Page fail (due to AR/ARx parity error?
                                        ;                 if yes - error
                                        ;                 if no - ok
                                        ;               Write 1 to CSR (should now have odd parity)
                                        ;               Read CSR
                                        ;               Page fail (due to AR/ARx parity error?
                                        ;                 if yes - error
                                        ;                 if no - ok
                                        ;
                                        ; Failure:      Parity not received properly over the EBUS.
                                        ;               KMUX Parity Generators
                                        ;#*****************************************************************

                                        ; Test data

004704' 254 00 0 00 004711'             TSTE54: JRST    TG54                    ; go start test
004705' 400000  000054                          EBUS!ZEBUS!54                   ; test mask
004706' 000000  012163'                         0,,[ASCIZ ^Basic KMUX Parity Test^]
004707' 011021' 011022'                         [EXP MLAST!E18],,[EXP E9,MLAST!E22]
004710' 777777  777777                          -1                              ; failure test table

                                        ; Start test

004711' 332 00 0 00 030037              TG54:   SKIPE   USER                    ; user mode?
                                                GO      [MOVE   TSTFLG          ; yes - ensure this test does not
                                                         TLO    (TUSER)         ;    get run (but only if not in
                                                         SKIPN  UDEBUG          ;    debug mode)
                                                         MOVEM  TSTFLG
004712' 260 17 0 00 010511'                              RTN]
004713' 201 00 0 00 000000'                     MOVEI   Z2                      ; get address of module start
004714' 260 17 0 00 004647*                     GO      TRACE                   ; handle trace output

004715' 400 15 0 00 000000              TA54:   SETZ    ERFLG,                  ; clear error flag
004716' 260 17 0 00 003547*                     GO      ERESET                  ; do an EBUS Reset
004717' 402 00 0 00 001035*                     SETZM   PFAIL                   ; clear page fail count
```

```

                                        ; First check for page fail

        004720' 260 17 0 00 004666*             GO      RDCSR                   ; read CSR
        004721' 255 00 0 00 000000              JFCL                            ; error - ignore
        004722' 332 00 0 00 004717*             SKIPE   PFAIL                   ; page fail occurred?
        004723' 254 00 0 00 004737'             JRST    TB54                    ; yes - go handle error

                                        ; 2nd check for page fail

        004724' 400 01 0 00 000000              SETZ    1,                      ; clear AC1
        004725' 260 17 0 00 004474*             GO      LDCSR                   ; write 0's to CSR
        004726' 260 17 0 00 004720*             GO      RDCSR                   ; read CSR
        004727' 255 00 0 00 000000              JFCL                            ; error - ignore
        004730' 332 00 0 00 004722*             SKIPE   PFAIL                   ; page fail occurred?
        004731' 254 00 0 00 004737'             JRST    TB54                    ; yes - go handle error

                                        ; 3rd check for page fail

        004732' 201 01 0 00 000001              MOVEI   1,1                     ; set up AC1
        004733' 260 17 0 00 004725*             GO      LDCSR                   ; write 1 to CSR
        004734' 260 17 0 00 004726*             GO      RDCSR                   ; read CSR
        004735' 255 00 0 00 000000              JFCL                            ; error - ignore
        004736' 332 00 0 00 004730*             SKIPE   PFAIL                   ; page fail occurred?
        004737' 474 15 0 00 000000      TB54:   SETO    ERFLG,                  ; yes - set error flag

                                        ; Error handling

        004740' 027 00 0 00 004744'             SCOPER  MA54                    ; print error message
        004741' 254 00 0 00 004715'             JRST    TA54                    ; loop on error
        004742' 254 00 0 00 004743'             JRST    TX54                    ; altmode exit

                                        ; End of test

        004743' 263 17 0 00 000000      TX54:   RTN                             ; return

                                        ; Error messages

        004744' 140000  012170'         MA54:   MSG!TXNOT![ASCIZ /Did EBUS Reset, then CONI CSR/]
        004745' 270000  004746'                 LAST!CALL!TXALL!MA54PN

        004746' 037 00 0 00 012176'     MA54PN: TMSGC   <Page fail occurred (AR/ARx parity error)>
        004747' 260 17 0 00 010314'             GO      PFPNT                   ; print page fail data
        004750' 263 17 0 00 000000              RTN                             ; return
```

```
                                    ;#*******************************************************************
                                    ;* TEST 55 - Clearing MPROC Run Bit
                                    ;
                                    ; Description:  This test verifies that the  'MPROC Run'  bit  is
                                    ;               actually is reset upon detection of an error.
                                    ;
                                    ; Procedure:    KL> EBUS Reset
                                    ;               KL> Load test ucode with bad parity
                                    ;               KL> Write CSR to set MPROC Run
                                    ;               UC> The first microword executes ok
                                    ;               UC> The 2nd microword generates CRAM PE
                                    ;               KL> Read CSR - MPROC Run should be cleared
                                    ;
                                    ; Failure:      MPROC Err sets whenever the micro-processor sets
                                    ;               either CRAM PE or MBUS Error.  This bit then
                                    ;               clears MPROC Run.
                                    ;#*******************************************************************

                                    ; Test data

004751' 254 00 0 00 004763'         TSTE55: JRST    TG55                    ; go start test
004752' 400400  000055                      EBUS!NDMP!ZEBUS!55              ; test mask
004753' 005016' 012207'                     T55M,,[ASCIZ ^Clearing MPROC Run Bit^]
004754' 012214' 000000                      [EXP E17,MLAST!E14],,0
004755' 000000  005025'                     TSTE56                          ; failure test table
004756' 000000000000#                       TSTU10                          ;   ...
004757' 000000000000#                       TSTU11
004760' 000000000000#                       TSTU12
004761' 000000000000#                       TSTU13
004762' 777777  777777                      -1

                                    ; Start test

004763' 201 00 0 00 000000'         TG55:   MOVEI   Z2                      ; get address of module start
004764' 260 17 0 00 004714*                 GO      TRACE                   ; handle trace output
004765' 201 01 0 00 005016'                 MOVEI   1,T55M                  ; set up microcode address
004766' 260 17 0 00 004651*                 GO      TLOAD                   ; load/verify it
004767' 263 17 0 00 000000                  RTN                             ; failed - exit test

                                    ; 1st segment of test (Segment A) (Start the port)

004770' 400 15 0 00 000000          TA55:   SETZ    ERFLG,                  ; clear error flag
004771' 201 00 0 00 000001                  MOVEI   1                       ; set subtest number
004772' 202 00 0 00 004665*                 MOVEM   TSTSUB                  ;   to 1
004773' 201 00 0 00 000010                  MOVEI   MPRUN                   ; only set MPRUN bit when
004774' 202 00 0 00 004657*                 MOVEM   SDATA                   ;    starting the port
004775' 402 00 0 00 004660*                 SETZM   SNEXT                   ; set start address to 0
004776' 260 17 0 00 010277'                 GO      MSTART                  ; start up port
004777' 254 00 0 00 004770'                 JRST    TA55                    ; loop on error
005000' 254 00 0 00 005013'                 JRST    TX55                    ; altmode exit
005001' 326 15 0 00 005013'                 JUMPN   ERFLG,TX55              ; error? yes - exit
```

```

                                     ; 2nd segment of test (Segment B) (Verify that MPRun cleared in CSR)

005002' 350 00 0 00 004772*          TB55:  AOS     TSTSUB                  ; point to next subtest
005003' 260 17 0 00 004734*                 GO      RDCSR                   ; read it
005004' 474 15 0 00 000000                  SETO    ERFLG,                  ; error
005005' 602 01 0 00 000010                  TRNE    1,MPRUN                 ; bit set?
005006' 474 15 0 00 000000                  SETO    ERFLG,                  ; yes - set error flag
005007' 202 01 0 00 010365'                 MOVEM   1,SAVCSR                ; save CSR data
005010' 027 00 0 00 005014'                 SCOPER  MB55                    ; print error message
005011' 254 00 0 00 004770'                 JRST    TA55                    ; loop on error
005012' 254 00 0 00 005013'                 JRST    TX55                    ; altmode exit

                                     ; End of test

005013' 263 17 0 00 000000           TX55:  RTN                             ; return

                                     ; Error message - MPRun did not clear

005014' 140000  012216'              MB55:  MSG!TXNOT![ASCIZ /CSR bit 32 'MPRun' not cleared after CRAM PE/]
005015' 270000  010231'                     LAST!CALL!TXALL!MPNT2

                                     ; Microcode:    Cause CRAM PE on 2nd microword

005016' 000000  010000               T55M:  MWORD   <ADDR=0,JMAP,J=1>                ; 0
005017' 000000  000040
005020' 000100  010000                      MWORD   <JMAP,J=1,BAD>                   ; 1
005021' 000000  000041
005022' 000200  020000                      MWORD   <JMAP,J=2>                       ; 2
005023' 000000  000040
005024' 777777  777777                      -1
```

```

                                      ;#***************************************************************
                                      ;* TEST 56 - Clearing MPROC Run Bit
                                      ;
                                      ; Description:   This test verifies that the  'MPROC Run'  bit  is
                                      ;                actually is reset when a clear port is done.
                                      ;
                                      ; Procedure:     KL> EBUS Reset
                                      ;                KL> Load test ucode
                                      ;                KL> Write CSR to set MPROC Run
                                      ;                KL> Write CSR to set 'Clear Port'
                                      ;                KL> Read CSR - MPROC Run should be cleared
                                      ;
                                      ; Failure:       Setting the 'Clear Port' bit should immediately
                                      ;                clear MPROC Run.
                                      ;#***************************************************************

                                      ; Test data

005025' 254 00 0 00 005036'           TSTE56: JRST    TG56                    ; go start test
005026' 400400  000056                        EBUS!NDMP!ZEBUS!56              ; test mask
005027' 005073' 012207'                       T56M,,[ASCIZ ^Clearing MPROC Run Bit^]
005030' 012227' 000000                        [EXP E17,MLAST!E6],,0
005031' 000000000000#                         TSTU10                          ; failure test table
005032' 000000000000#                         TSTU11                          ;    ...
005033' 000000000000#                         TSTU12
005034' 000000000000#                         TSTU13
005035' 777777  777777                        -1

                                      ; Start test

005036' 201 00 0 00 000000'           TG56:   MOVEI   Z2                      ; get address of module start
005037' 260 17 0 00 004764*                   GO      TRACE                   ; handle trace output
005040' 201 01 0 00 005073'                   MOVEI   1,T56M                  ; set up microcode address
005041' 260 17 0 00 004766*                   GO      TLOAD                   ; load/verify it
005042' 263 17 0 00 000000                    RTN                             ; failed - exit test

                                      ; 1st segment of test (Segment A) (Start the port)

005043' 400 15 0 00 000000            TA56:   SETZ    ERFLG,                  ; clear error flag
005044' 201 00 0 00 000001                    MOVEI   1                       ; set subtest number
005045' 202 00 0 00 005002*                   MOVEM   TSTSUB                  ;   to 1
005046' 201 00 0 00 000010                    MOVEI   MPRUN                   ; only set MPRUN bit when
005047' 202 00 0 00 004774*                   MOVEM   SDATA                   ;    starting the port
005050' 402 00 0 00 004775*                   SETZM   SNEXT                   ; set start address to 0
005051' 260 17 0 00 010277'                   GO      MSTART                  ; start up port
005052' 254 00 0 00 005043'                   JRST    TA56                    ; loop on error
005053' 254 00 0 00 005070'                   JRST    TX56                    ; altmode exit
005054' 326 15 0 00 005070'                   JUMPN   ERFLG,TX56              ; error? yes - exit
```

```

                                        ; 2nd segment of test (Segment B) (Verify that MPRun cleared in CSR)

005055' 350 00 0 00 005045*     TB56:   AOS     TSTSUB                  ; point to next subtest
005056' 201 01 0 00 400010              MOVEI   1,PCLEAR!MPRUN          ; set both bits
005057' 260 17 0 00 004733*             GO      LDCSR                   ; write them
005060' 260 17 0 00 005003*             GO      RDCSR                   ; read it
005061' 474 15 0 00 000000              SETO    ERFLG,                  ; error
005062' 602 01 0 00 000010              TRNE    1,MPRUN                 ; bit set?
005063' 474 15 0 00 000000              SETO    ERFLG,                  ; yes - set error flag
005064' 202 01 0 00 010365'             MOVEM   1,SAVCSR                ; save CSR data
005065' 027 00 0 00 005071'             SCOPER  MB56                    ; print error message
005066' 254 00 0 00 005043'             JRST    TA56                    ; loop on error
005067' 254 00 0 00 005070'             JRST    TX56                    ; altmode exit

                                        ; End of test

005070' 263 17 0 00 000000      TX56:   RTN                             ; return

                                        ; Error message - MPRun did not clear

005071' 140000  012231'         MB56:   MSG!TXNOT![ASCIZ /CSR bit 32 'MPRun' not cleared after 'Port Clear'/]
005072' 270000  010231'                 LAST!CALL!TXALL!MPNT2

                                        ; Microcode:    Infinite loop.

005073' 000000  000000          T56M:   MWORD   <ADDR=0,JZ>
005074' 000000  000000
005075' 777777  777777                  -1
```

```

                                     ;#*************************************************************************
                                     ;* TEST 57 - Verify 'Diag Single Cyc'
                                     ;
                                     ; Description:  This test  verifies  when CSR Bit 22 'Diag Single
                                     ;               Cycle' is set, the microprocessor stops after one
                                     ;               instruction and clears 'MPROC Run' bit.
                                     ;
                                     ; Procedure:    KL> EBUS Reset
                                     ;               KL> Load test ucode
                                     ;               KL> Write CSR - set 'Single Cycle' and 'MPROC Run'
                                     ;               KL> Read CSR - MPROC Run should be cleared
                                     ;                      if not - error
                                     ;               KL> 'Single Cycle' bit still set?
                                     ;                      if not - error
                                     ;
                                     ; Failure:      ---
                                     ;#*************************************************************************

                                     ; Test data

005076' 254 00 0 00 005103'          TSTE57: JRST    TG57                          ; go start test
005077' 400400  000057                       EBUS!NDMP!ZEBUS!57                    ; test mask
005100' 005140' 012243'                      T57M,,[ASCIZ ^Verify 'Diag Single Cyc'^]
005101' 012250' 012253'                      [EXP E17,E12,MLAST!M1],,[EXP E6,E14,MLAST!E1]
005102' 777777  777777                       -1                                    ; failure test table

                                     ; Start test

005103' 201 00 0 00 000000'          TG57:   MOVEI   Z2                            ; get address of module start
005104' 260 17 0 00 005037*                  GO      TRACE                         ; handle trace output
005105' 201 01 0 00 005140'                  MOVEI   1,T57M                        ; set up microcode address
005106' 260 17 0 00 005041*                  GO      TLOAD                         ; load/verify it
005107' 263 17 0 00 000000                   RTN                                   ; failed - exit test

                                     ; 1st segment of test (Segment A) (Start the port)

005110' 400 15 0 00 000000           TA57:   SETZ    ERFLG,                        ; clear error flag
005111' 201 00 0 00 000001                   MOVEI   1                             ; set subtest number
005112' 202 00 0 00 005055*                  MOVEM   TSTSUB                        ;   to 1
005113' 201 00 0 00 000010                   MOVEI   MPRUN                         ; only set MPRUN bit when
005114' 202 00 0 00 005047*                  MOVEM   SDATA                         ;      starting the port
005115' 402 00 0 00 005050*                  SETZM   SNEXT                         ; set start address to 0
005116' 260 17 0 00 010277'                  GO      MSTART                        ; start up port
005117' 254 00 0 00 005110'                  JRST    TA57                          ; loop on error
005120' 254 00 0 00 005135'                  JRST    TX57                          ; altmode exit
005121' 326 15 0 00 005135'                  JUMPN   ERFLG,TX57                    ; error? yes - exit
```

```

                                           ; 2nd segment of test (Segment B) (Verify that MPRun cleared in CSR)

   005122' 350 00 0 00 005112*        TB57:  AOS      TSTSUB                      ; point to next subtest
   005123' 201 01 0 00 020010                MOVEI    1,SINCYC!MPRUN              ; set both bits
   005124' 260 17 0 00 005057*               GO       LDCSR                       ; write them
   005125' 260 17 0 00 005060*               GO       RDCSR                       ; read it
   005126' 474 15 0 00 000000                SETO     ERFLG,                      ; error
   005127' 602 01 0 00 000010                TRNE     1,MPRUN                     ; bit set?
   005130' 474 15 0 00 000000                SETO     ERFLG,                      ; yes - set error flag
   005131' 202 01 0 00 010365'               MOVEM    1,SAVCSR                    ; save CSR data
   005132' 027 00 0 00 005136'               SCOPER   MB57                        ; print error message
   005133' 254 00 0 00 005110'               JRST     TA57                        ; loop on error
   005134' 254 00 0 00 005135'               JRST     TX57                        ; altmode exit

                                      ; End of test

   005135' 263 17 0 00 000000         TX57:  RTN                                  ; return

                                      ; Error message - MPRun did not clear

   005136' 140000  012256'            MB57:  MSG!TXNOT![ASCIZ /CSR bit 32 'MPRun' not cleared after a single cycle/]
   005137' 270000  010231'                   LAST!CALL!TXALL!MPNT2

                                      ; Microcode:     Infinite loop.

   005140' 000000  000000             T57M:  MWORD    <ADDR=0,JZ>
   005141' 000000  000000
   005142' 777777  777777                    -1
```

```
                                ;#*********************************************************************
                                ;* TEST 60 - PI Level n/0 Interrupts
                                ;
                                ; Description:   This test verifies that  microcode  can  perform
                                ;                PI Level 0 interrupts at the same time that a PI
                                ;                Level n interrupt is in progress.
                                ;
                                ; Procedure:     KL> Port Clear
                                ;                KL> Load test ucode
                                ;                KL> Initialize PI system, but leave interrupts off
                                ;                KL> Clear locations BUFF => BUFF+17
                                ;                KL> Write CSR to set MPROC Run (PIA bits = 0)
                                ;                KL> Write physical addr BUFF to port
                                ;                KL> Set CSR Register PIA bits 33-35 to 4
                                ;
                                ;                KL> Wait for non-vectored interrupt
                                ;                KL> Service and dismiss interrupt (PI system shut
                                ;                    off for anywhere between 20 ms)
                                ;
                                ;                KL> Verify the number of increments done, within
                                ;                    an acceptable range.
                                ;                If error - print error message
                                ;
                                ; Failure:       ---
                                ;#*********************************************************************

                                ; Test data

005143' 254 00 0 00 005150'     TSTE60: JRST    TG60                    ; go start test
005144' 400400  000060                  EBUS!NDMP!ZEBUS!60              ; test mask
005145' 005233' 012271'                 T60M,,[ASCIZ ^PI Level n/0 Interrupts^]
005146' 012276' 012300'                 [EXP E14,MLAST!E7],,[EXP E4,MLAST!E1]
005147' 777777  777777                  -1

                                ; Start test

005150' 332 00 0 00 030037      TG60:   SKIPE   USER                    ; user mode?
                                        GO      [MOVE   TSTFLG          ; yes - ensure this test does not
                                                 TLO    (TUSER)         ;     get run (but only if not in
                                                 SKIPN  UDEBUG          ;     debug mode)
                                                 MOVEM  TSTFLG
005151' 260 17 0 00 010511'                      RTN]
005152' 201 00 0 00 000000'             MOVEI   Z2                      ; get address of module start
005153' 260 17 0 00 005104*             GO      TRACE                   ; handle trace output
005154' 201 01 0 00 005233'             MOVEI   1,T60M                  ; set up microcode address
005155' 260 17 0 00 005106*             GO      TLOAD                   ; load/verify it
005156' 263 17 0 00 000000              RTN                             ; failed - exit test
```

```

                                        ; Initialization

005157' 400 15 0 00 000000      TL60:   SETZ    ERFLG,                  ; clear error flag
005160' 402 00 0 00 000000*             SETZM   BUFF                    ; clear buffer area
005161' 200 00 0 00 012302'             MOVE    [BUFF,,BUFF+1]          ;   ...
005162' 251 00 0 00 000000#             BLT     BUFF+^D100              ;   ...
005163' 260 17 0 00 004507*             GO      IPACLR                  ; do a 'port clear'
005164' 260 17 0 00 000000*             GO      INITPI                  ; init PI system
005165' 260 17 0 00 000000*             GO      SETVEC                  ; set up vector addresses
005166' 260 17 0 00 000000*             GO      PISYON                  ; turn on PI system
005167' 260 17 0 00 000000*             GO      PISYOF                  ; turn off PI system

                                        ; Start up the port

005170' 201 01 0 00 005000              MOVEI   1,5000                  ; get start address (5000)
005171' 242 01 0 00 000001              LSH     1,1                     ; position properly
005172' 260 17 0 00 003376*             GO      LDRAR                   ; load the register
005173' 201 01 0 00 000014              MOVEI   1,MPRUN!4               ; set initial data
005174' 260 17 0 00 005124*             GO      LDCSR                   ; write to CSR
005175' 201 02 0 00 005160*             MOVEI   2,BUFF                  ; get buffer address
005176' 336 00 0 00 003406*             SKIPN   UDEBUG                  ; debug mode?
005177' 257 02 0 02 000000              MAP     2,(2)                   ; no - map to physical
005200' 621 02 0 00 777000              TLZ     2,777000                ; mask off unused bits
005201' 200 01 0 00 012303'             MOVE    1,[DATAO 2]             ; build a DATAO
005202' 434 01 0 00 000016              IOR     1,MBCN                  ; include the device code
005203' 336 00 0 00 005176*             SKIPN   UDEBUG                  ; debug mode?
005204' 256 00 0 00 000001              XCT     1                       ; no - do the DATAO

                                        ; Wait a bit

005205' 201 00 0 00 000024              MOVEI   ^D20                    ; wait 20 milliseconds
005206' 260 17 0 00 000000*             GO      ODELAY                  ;  ...

                                        ; Turn on PI system and wait for interrupt

005207' 260 17 0 00 005166*             GO      PISYON                  ; turn on PI system
005210' 201 01 0 00 000764              MOVEI   1,^D500                 ; check only 500 times
005211' 335 00 0 00 000000*             SKIPGE  INTNUM                  ; interrupt occurred yet?
005212' 367 01 0 00 005211'             SOJG    1,.-1                   ; no - loop till one does
005213' 260 17 0 00 005163*             GO      IPACLR                  ; do a 'port clear'

                                        ; Check results

005214' 200 00 0 00 005175*             MOVE    BUFF                    ; get number of examines done
005215' 305 00 0 00 002734              CAIGE   ^D1500                  ; within range?
005216' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
```

```

                                        ; Handle error printouts and scope looping

005217' 027 00 0 00 005223'                     SCOPER  MA60                    ; print error message
005220' 254 00 0 00 005157'                     JRST    TL60                    ; loop on error
005221' 254 00 0 00 005222'                     JRST    TX60                    ; altmode exit

                                        ; End of test

005222' 263 17 0 00 000000              TX60:   RTN                             ; return

                                        ; Error messages

005223' 270000  005224'                 MA60:   LAST!CALL!TXALL!MA60PN          ; print test data

                                        MA60PN: TMSG    <
                                        While PI Level 4 interrupt requested but PI system off, more than 1500.
005224' 037 00 0 00 012304'               interrupts should have occurred - only >
005225' 200 00 0 00 005214*                     MOVE    BUFF
005226' 037 15 0 00 000000                      PNTDEC
005227' 037 00 0 00 012334'                     TMSG    <. actually did.>
005230' 335 00 0 00 005211*                     SKIPGE  INTNUM                  ; an interrupt occurred?
005231' 037 00 0 00 012340'                     TMSGC   <PI Level 4 interrupt did not occur when PI system back on>
005232' 263 17 0 00 000000                      RTN

                                        ; Microcode:

005233' 500000  000000                  T60M:   MWORD   <ADDR=5000,JZ>                          ; 5000
005234' 000000  000000

                                        ;       Initialization

005235' 000000  010000                          MWORD   <ADDR=0,JMAP,J=1>                       ; 0
005236' 000000  000040
005237' 000100  030000                          MWORD   <CJP,J=3,CENA,CCER>                     ; 1
005240' 000400  100060
005241' 000200  010000                          MWORD   <JMAP,J=1>                              ; 2
005242' 000000  000040
005243' 000300  000010                          MWORD   <CONT,SDO,OR,B=5,D=2,SELE,MGC=10>       ; 3
005244' 732002  405340
005245' 000400  320470                          MWORD   <LDCT,J=32,SDO,OR,B=7,D=2,SKCN,MGC=470> ; 4
005246' 732003  640300
005247' 000500  050000                          MWORD   <RPCT,J=5,SOB,OR,B=7,D=7>               ; 5
005250' 337003  400220
005251' 000600  310020                          MWORD   <LDCT,J=31,SDO,OR,B=11,D=2,SKCN,MGC=20> ; 6
005252' 732004  640300
005253' 000700  070000                          MWORD   <RPCT,J=7,SOB,OR,B=11,D=5>              ; 7
005254' 335004  400220
005255' 001000  120040                          MWORD   <CJP,J=12,CENA,CCGC,SELE,MGC=40>        ; 10
005256' 000400  015060
005257' 001100  100000                          MWORD   <JMAP,J=10>                             ; 11
005260' 000000  000040
005261' 001200  200100                          MWORD   <JMAP,J=20,SDO,B=2,OR,D=2,SELE,MGC=100> ; 12
005262' 732001  005040

```

```
4913                                    ;       Request a PI level n interrupt
4914
4915  005263' 002000  210002                  MWORD   <ADDR=20,JMAP,J=21,SOA,AND,B=6,D=2,SELE,MGC=2> ; 20
4916  005264' 442003  005040
4917  005265' 002101  000000                  MWORD   <CJS,J=100,SOB,PLUS,B=6,D=2,CRY>       ; 21
4918  005266' 302003  000420
4919  005267' 002200  210000                  MWORD   <CJP,J=21,CENA,CCIA>                   ; 22
4920  005270' 000400  110060
4921  005271' 002300  230000                  MWORD   <JMAP,J=23>                            ; 23
4922  005272' 000000  000040
4923
4924                                    ;       Subroutine to do a PI level 0 interrupt (Examine/increment)
4925
4926  005273' 010001  010000                  MWORD   <ADDR=100,JMAP,J=101,SOA,OR,A=5,B=10,D=2> ; 100
4927  005274' 432054  000040
4928  005275' 010100  050000                  MWORD   <LDCT,J=5,D=2>                         ; 101
4929  005276' 002000  000300
4930  005277' 010201  020000                  MWORD   <RPCT,J=102>                           ; 102
4931  005300' 000000  000220
4932  005301' 010300  000000                  MWORD   <CONT,SAB,OR,A=7,B=10,D=2>             ; 103
4933  005302' 132074  000340
4934  005303' 010400  002004                  MWORD   <CONT,SOB,OR,B=10,D=1,OENA,SELE,MGC=4> ; 104
4935  005304' 331004  005340
4936  005305' 010500  000001                  MWORD   <CONT,SELE,MGC=1>                      ; 105
4937  005306' 000000  005340
4938  005307' 010601  100000                  MWORD   <CJP,J=110,CENA,CCER>                  ; 106
4939  005310' 000400  100060
4940  005311' 010701  060000                  MWORD   <JMAP,J=106>                           ; 107
4941  005312' 000000  000040
4942  005313' 011000  000010                  MWORD   <CONT,SDO,OR,D=2,B=16,SELE,MGC=10>     ; 110
4943  005314' 732007  005340
4944  005315' 011100  000000                  MWORD   <CRTN>                                 ; 111
4945  005316' 000000  000240
4946  005317' 777777  777777                  -1
```

```

                                        ;#********************************************************************
                                        ;* TEST 61 - Check PIA Level 0
                                        ;
                                        ; Description:  This test verifies when CSR Bit 33-35 specify PIA
                                        ;               Level 0, that no interrupt occurs.  The PI system
                                        ;               is turned  on  so that interrupts would otherwise
                                        ;               occur.
                                        ;
                                        ; Procedure:    KL> EBUS Reset
                                        ;               KL> Load test ucode (with bad parity)
                                        ;               KL> Set up PI system so that interrupts are handled
                                        ;                   properly.
                                        ;               KL> Set CSR Register PIA bits 33-35 to 0
                                        ;               KL> Write CSR to set MPROC Run
                                        ;               KL> Wait for interrupt, and if one occurs, print
                                        ;                   error information.
                                        ;
                                        ; Failure:      ---
                                        ;#********************************************************************

                                        ; Test data

005320' 254 00 0 00 005327'             TSTE61: JRST    TG61                    ; go start test
005321' 400400  000061                          EBUS!NDMP!ZEBUS!61              ; test mask
005322' 005372' 012354'                         T61M,,[ASCIZ ^Check PIA Level 0^]
005323' 011570' 012360'                         [EXP MLAST!E4],,[EXP E1,E14,E15,MLAST!E6]
005324' 000000  005401'                         TSTE62                          ; failure test table
005325' 000000  005503'                         TSTE63                          ;   ...
005326' 777777  777777                          -1

                                        ; Start test

005327' 332 00 0 00 030037              TG61:   SKIPE   USER                    ; user mode?
                                                GO      [MOVE   TSTFLG          ; yes - ensure this test does not
                                                         TLO    (TUSER)         ;    get run (but only if not in
                                                         SKIPN  UDEBUG          ;    debug mode)
                                                         MOVEM  TSTFLG
005330' 260 17 0 00 010511'                              RTN]
005331' 201 00 0 00 000000'                     MOVEI   Z2                      ; get address of module start
005332' 260 17 0 00 005153*                     GO      TRACE                   ; handle trace output
005333' 201 01 0 00 005372'                     MOVEI   1,T61M                  ; set up microcode address
005334' 260 17 0 00 005155*                     GO      TLOAD                   ; load/verify it
005335' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Set up interrupt system, etc.

005336' 400 15 0 00 000000              TA61:   SETZ    ERFLG,                  ; clear error flag
005337' 260 17 0 00 005213*                     GO      IPACLR                  ; do a 'port clear'
005340' 260 17 0 00 005164*                     GO      INITPI                  ; init PI system
005341' 260 17 0 00 005165*                     GO      SETVEC                  ; set up vector addresses
005342' 260 17 0 00 000000*                     GO      .PION                   ; set up CSR ok and turn on PI
```

```

                                    ; 1st segment of test (Segment A) (Start the port)

005343' 201 00 0 00 000001                  MOVEI   1                           ; set subtest number
005344' 202 00 0 00 005122*                 MOVEM   TSTSUB                      ;   to 1
005345' 201 00 0 00 000010                  MOVEI   MPRUN                       ; get startup bits (PI level 0)
005346' 202 00 0 00 005114*                 MOVEM   SDATA                       ; save in startup data word
005347' 402 00 0 00 005115*                 SETZM   SNEXT                       ; set start address to 0
005350' 260 17 0 00 010277'                 GO      MSTART                      ; start up port
005351' 254 00 0 00 005336'                 JRST    TA61                        ; loop on error
005352' 254 00 0 00 005366'                 JRST    TX61                        ; altmode exit
005353' 326 15 0 00 005366'                 JUMPN   ERFLG,TX61                  ; error? yes - exit

                                    ; 2nd segment of test (Segment B) (Look for interrupt)

005354' 350 00 0 00 005344*         TB61:   AOS     TSTSUB                      ; point to next subtest
005355' 201 00 0 00 000012                  MOVEI   ^D10                        ; wait 10 ms for interrupt
005356' 260 17 0 00 005206*                 GO      ODELAY                      ; do the delay
005357' 331 00 0 00 005230*                 SKIPL   INTNUM                      ; any interrupt occurred?
005360' 474 15 0 00 000000                  SETO    ERFLG,                      ; yes - flag error
005361' 260 17 0 00 005125*                 GO      RDCSR                       ; read CSR
005362' 255 00 0 00 000000                  JFCL                                ; error - ignore it
005363' 027 00 0 00 005367'                 SCOPER  MB61                        ; print error message
005364' 254 00 0 00 005336'                 JRST    TA61                        ; loop on error
005365' 254 00 0 00 005366'                 JRST    TX61                        ; altmode exit

                                    ; End of test

005366' 263 17 0 00 000000          TX61:   RTN                                 ; return

                                    ; Error message - An interrupt request seen

005367' 140000  012364'             MB61:   MSG!TXNOT![ASCIZ /An interrupt occurred with PI Level 0/]
005370' 260000  010231'                     CALL!TXALL!MPNT2
005371' 270000  010331'                     LAST!CALL!TXALL!INTPNT

                                    ; Microcode:    Cause cram parity error.

005372' 000000  010000              T61M:   MWORD   <ADDR=0,JMAP,J=1>                     ; 0
005373' 000000  000040
005374' 000100  010000                      MWORD   <JMAP,J=1,BAD>                        ; 1
005375' 000000  000041
005376' 000200  020000                      MWORD   <JMAP,J=2>                            ; 2
005377' 000000  000040
005400' 777777  777777                      -1
```

```

                                ;#*******************************************************************
                                ;* TEST 62 - Check Interrupt on Level 1-7
                                ;
                                ; Description:  This test verifies when CSR Bit 33-35 specify PIA
                                ;               Levels 1-7 and an interrupt should occur, that an
                                ;               interrupt does in fact occur and on the specified
                                ;               level only.
                                ;
                                ; Procedure:    KL> EBUS Reset
                                ;               KL> Load test ucode (with bad parity)
                                ;               KL> Set up PI system so that no interrupts occur
                                ;                   but they are recognized
                                ;               KL> Set CSR Register PIA bits 33-35 to 1
                                ;               KL> Write CSR to set MPROC Run
                                ;               KL> Do a CONI PI - check that the proper program
                                ;                   request bit is set
                                ;               KL> Do a CONO PI to enable the interrupt request
                                ;                   on all levels
                                ;               KL> Process the interrupt and verify that only
                                ;                   one occurred and on the proper level
                                ;
                                ;               Repeat for PI Levels 2-7
                                ;
                                ; Failure:      ---
                                ;#*******************************************************************

                                ; Test data

005401' 254 00 0 00 005411'     TSTE62: JRST    TG62                    ; go start test
005402' 400400  000062                  EBUS!NDMP!ZEBUS!62              ; test mask
005403' 005474' 012374'                 T62M,,[ASCIZ ^Check Interrupt on Level 1-7^]
005404' 011570' 012360'                 [EXP MLAST!E4],,[EXP E1,E14,E15,MLAST!E6]
005405' 000000  005503'                 TSTE63                          ; failure test table
005406' 000000  005603'                 TSTE64                          ;   ...
005407' 000000  006030'                 TSTE65
005410' 777777  777777                  -1

                                ; Start test

005411' 332 00 0 00 030037      TG62:   SKIPE   USER                    ; user mode?
                                        GO      [MOVE   TSTFLG          ; yes - ensure this test does not
                                                 TLO    (TUSER)         ;    get run (but only if not in
                                                 SKIPN  UDEBUG          ;    debug mode)
                                                 MOVEM  TSTFLG
005412' 260 17 0 00 010511'                      RTN]
005413' 201 00 0 00 000000'             MOVEI   Z2                      ; get address of module start
005414' 260 17 0 00 005332*             GO      TRACE                   ; handle trace output
005415' 201 01 0 00 005474'             MOVEI   1,T62M                  ; set up microcode address
005416' 260 17 0 00 005334*             GO      TLOAD                   ; load/verify it
005417' 263 17 0 00 000000              RTN                             ; failed - exit test
```

```

                                          ; Initialize PI level

005420' 402 00 0 00 010361'                       SETZM   SAVDAT                   ; set initial PI level to
005421' 350 00 0 00 010361'                       AOS     SAVDAT                   ;   be 1

                                          ; Set up interrupt system, etc.

005422' 400 15 0 00 000000                TA62:   SETZ    ERFLG,                   ; clear error flag
005423' 260 17 0 00 005340*                       GO      INITPI                   ; init PI system
005424' 260 17 0 00 005341*                       GO      SETVEC                   ; set up vector addresses
005425' 260 17 0 00 005342*                       GO      .PION                    ; set up CSR ok and turn on PI

                                          ; 1st segment of test (Segment A) (Start the port)

005426' 200 00 0 00 010361'                       MOVE    SAVDAT                   ; set subtest number to
005427' 242 00 0 00 000001                        LSH     1                        ;   PI level * 2 -1
005430' 275 00 0 00 000001                        SUBI    1                        ;   and save
005431' 202 00 0 00 005354*                       MOVEM   TSTSUB
005432' 201 00 0 00 000010                        MOVEI   MPRUN                    ; get startup bits
005433' 434 00 0 00 010361'                       OR      SAVDAT                   ; insert PI level
005434' 202 00 0 00 005346*                       MOVEM   SDATA                    ; save in startup data word
005435' 402 00 0 00 005347*                       SETZM   SNEXT                    ; set start address to 0
005436' 260 17 0 00 010277'                       GO      MSTART                   ; start up port
005437' 254 00 0 00 005422'                       JRST    TA62                     ; loop on error
005440' 254 00 0 00 005464'                       JRST    TX62                     ; altmode exit
005441' 326 15 0 00 005464'                       JUMPN   ERFLG,TX62               ; error? yes - exit

                                          ; 2nd segment of test (Segment B) (Field interrupt)

005442' 201 02 0 00 000620                TB62:   MOVEI   2,^D400                  ; check up to 400 times
005443' 350 00 0 00 005431*                       AOS     TSTSUB                   ; point to next subtest
005444' 335 00 0 00 005357*               TB62A:  SKIPGE  INTNUM                   ; interrupt occurred yet?
005445' 367 02 0 00 005444'                       SOJG    2,TB62A                  ; no - loop till timeout
005446' 332 00 0 00 005444*                       SKIPE   INTNUM                   ; only one interrupt occurred?
005447' 474 15 0 00 000000                        SETO    ERFLG,                   ; no - set error flag
005450' 200 00 0 00 000000*                       MOVE    INTTYP                   ; get interrupt type
005451' 312 00 0 00 010361'                       CAME    SAVDAT                   ; non-vectored on level 4?
005452' 474 15 0 00 000000                        SETO    ERFLG,                   ; no - set error flag
005453' 260 17 0 00 005361*                       GO      RDCSR                    ; read CSR
005454' 255 00 0 00 000000                        JFCL                             ; error - ignore it
005455' 027 00 0 00 005465'                       SCOPER  MB62                     ; print error message
005456' 254 00 0 00 005422'                       JRST    TA62                     ; loop on error
005457' 254 00 0 00 005464'                       JRST    TX62                     ; altmode exit
005460' 326 15 0 00 005464'                       JUMPN   ERFLG,TX62               ; error? yes - exit

                                          ; Now do next PI level

005461' 350 01 0 00 010361'                       AOS     1,SAVDAT                 ; increment PI level
005462' 307 01 0 00 000007                        CAIG    1,7                      ; done all 7 yet?
005463' 254 00 0 00 005422'                       JRST    TA62                     ; no - loop till done
```

```

                                        ; End of test

005464' 263 17 0 00 000000      TX62:   RTN                                     ; return

                                        ; Error message - No interrupt seen

005465' 240000  005470'         MB62:   CALL!TXNOT!MB62PN
005466' 260000  010231'                 CALL!TXALL!MPNT2
005467' 270000  010331'                 LAST!CALL!TXALL!INTPNT

005470' 037 00 0 00 012402'     MB62PN: TMSGC   <Failed to get 1 non-vectored interrupt on Level >
005471' 200 00 0 00 010361'             MOVE    SAVDAT
005472' 037 16 0 00 000003              PNTOCS
005473' 263 17 0 00 000000              RTN

                                        ; Microcode:    Cause cram parity error.

005474' 000000  010000          T62M:   MWORD   <ADDR=0,JMAP,J=1>                       ; 0
005475' 000000  000040
005476' 000100  010000                  MWORD   <JMAP,J=1,BAD>                          ; 1
005477' 000000  000041
005500' 000200  020000                  MWORD   <JMAP,J=2>                              ; 2
005501' 000000  000040
005502' 777777  777777                  -1
```

```

                                ;#*********************************************************************
                                ;* TEST 63 - CSR 05 'Rqst Interrupt' Clearing
                                ;
                                ; Description:  Verify  that  CSR  bit  0   is  cleared  when  an
                                ;               interrupt occurs.
                                ;
                                ; Procedure:    KL> EBUS Reset
                                ;               KL> Load test ucode
                                ;               KL> Set up PI system so that interrupts are
                                ;                     recognized and enabled on Level 4
                                ;               KL> Set CSR Register PIA bits 33-35 to 4
                                ;               KL> Write CSR to set MPROC Run
                                ;               UC> Set MPRQSTINTR
                                ;               KL> Process the interrupt
                                ;               KL> Read the CSR and verify that CSR 05 'Rqst
                                ;                   Interrupt' cleared - error if it didn't.
                                ;
                                ; Failure:      When an interrupt occurs, the bit gets cleared
                                ;               by the interrupt handshake sequence (when an IOP
                                ;               word is put on the EBUS).
                                ;#*********************************************************************

                                ; Test data

005503' 254 00 0 00 005512'     TSTE63: JRST    TG63                    ; go start test
005504' 400400  000063                  EBUS!NDMP!ZEBUS!63              ; test mask
005505' 005574' 012415'                 T63M,,[ASCIZ ^CSR 05 'Rqst Interrupt' Clearing^]
005506' 011707' 012300'                 [EXP MLAST!E14],,[EXP E4,MLAST!E1]
005507' 000000  005603'                 TSTE64                          ; failure test table
005510' 000000  006030'                 TSTE65                          ;   ...
005511' 777777  777777                  -1

                                ; Start test

005512' 332 00 0 00 030037      TG63:   SKIPE   USER                    ; user mode?
                                        GO      [MOVE   TSTFLG          ; yes - ensure this test does not
                                                 TLO    (TUSER)         ;    get run (but only if not in
                                                 SKIPN  UDEBUG          ;    debug mode)
                                                 MOVEM  TSTFLG
005513' 260 17 0 00 010511'                      RTN]
005514' 201 00 0 00 000000'             MOVEI   Z2                      ; get address of module start
005515' 260 17 0 00 005414*             GO      TRACE                   ; handle trace output
005516' 201 01 0 00 005574'             MOVEI   1,T63M                  ; set up microcode address
005517' 260 17 0 00 005416*             GO      TLOAD                   ; load/verify it
005520' 263 17 0 00 000000              RTN                             ; failed - exit test
```

```

                                       ; Set up interrupt system, etc.

005521' 400 15 0 00 000000      TA63:  SETZ    ERFLG,                  ; clear error flag
005522' 260 17 0 00 005423*            GO      INITPI                  ; init PI system
005523' 260 17 0 00 005424*            GO      SETVEC                  ; set up vector addresses
005524' 260 17 0 00 005425*            GO      .PION                   ; set up CSR ok and turn on PI

                                ; 1st segment of test (Segment A) (Start the port)

005525' 201 00 0 00 000001             MOVEI   1                       ; set subtest number
005526' 202 00 0 00 005443*            MOVEM   TSTSUB                  ;   to 1
005527' 201 00 0 00 000014             MOVEI   MPRUN!4                 ; get startup bits
005530' 202 00 0 00 005434*            MOVEM   SDATA                   ; save in startup data word
005531' 402 00 0 00 005435*            SETZM   SNEXT                   ; set start address to 0
005532' 260 17 0 00 010277'            GO      MSTART                  ; start up port
005533' 254 00 0 00 005521'            JRST    TA63                    ; loop on error
005534' 254 00 0 00 005566'            JRST    TX63                    ; altmode exit
005535' 326 15 0 00 005566'            JUMPN   ERFLG,TX63              ; error? yes - exit

                                ; 2nd segment of test (Segment B) (Field interrupt)

005536' 201 02 0 00 000620             MOVEI   2,^D400                 ; check up to 400 times
005537' 350 00 0 00 005526*            AOS     TSTSUB                  ; point to next subtest
005540' 335 00 0 00 005446*     TB63:  SKIPGE  INTNUM                  ; interrupt occurred yet?
005541' 367 02 0 00 005540'            SOJG    2,TB63                  ; no - loop till timeout
005542' 332 00 0 00 005540*            SKIPE   INTNUM                  ; only one interrupt occurred?
005543' 474 15 0 00 000000             SETO    ERFLG,                  ; no - set error flag
005544' 200 00 0 00 005450*            MOVE    INTTYP                  ; get interrupt type
005545' 302 00 0 00 000004             CAIE    4                       ; non-vectored on level 4?
005546' 474 15 0 00 000000             SETO    ERFLG,                  ; no - set error flag
005547' 260 17 0 00 005453*            GO      RDCSR                   ; read CSR
005550' 255 00 0 00 000000             JFCL                            ; error - ignore it
005551' 027 00 0 00 005567'            SCOPER  MB63                    ; print error message
005552' 254 00 0 00 005521'            JRST    TA63                    ; loop on error
005553' 254 00 0 00 005566'            JRST    TX63                    ; altmode exit
005554' 326 15 0 00 005566'            JUMPN   ERFLG,TX63              ; error? yes - exit

                                ; 3rd segment of test (Segment C) (Verify 'Rqst Interrupt' bit cleared)

005555' 350 00 0 00 005537*     TC63:  AOS     TSTSUB                  ; point to next subtest
005556' 260 17 0 00 005547*            GO      RDCSR                   ; read CSR
005557' 474 15 0 00 000000             SETO    ERFLG,                  ; error
005560' 202 01 0 00 010365'            MOVEM   1,SAVCSR                ; save data
005561' 603 01 0 00 010000             TLNE    1,(RQINT)               ; 'Rqst Interrupt' still set?
005562' 474 15 0 00 000000             SETO    ERFLG,                  ; no - flag error
005563' 027 00 0 00 005572'            SCOPER  MC63                    ; print error message
005564' 254 00 0 00 005521'            JRST    TA63                    ; loop on error
005565' 254 00 0 00 005566'            JRST    TX63                    ; altmode exit

                                ; End of test

005566' 263 17 0 00 000000      TX63:  RTN                             ; return
```

```
5270
5271                                  ; Error message - No interrupt seen
5272
5273  005567' 140000  012424'         MB63:   MSG!TXNOT![ASCIZ /Failed to get 1 non-vectored interrupt on Level 4/]
5274  005570' 260000  010231'                 CALL!TXALL!MPNT2
5275  005571' 270000  010331'                 LAST!CALL!TXALL!INTPNT
5276
5277                                  ; Error message - 'Rqst Int' still set
5278
5279  005572' 140000  012436'         MC63:   MSG!TXNOT![ASCIZ /CSR Bit 05 'Rqst Int' still set/]
5280  005573' 270000  010231'                 LAST!CALL!TXALL!MPNT2
5281
5282                                  ; Microcode:    Cause cram parity error.
5283
5284  005574' 000000  010000          T63M:   MWORD   <ADDR=0,JMAP,J=1>                       ; 0
5285  005575' 000000  000040
5286  005576' 000100  010000                  MWORD   <JMAP,J=1,BAD>                          ; 1
5287  005577' 000000  000041
5288  005600' 000200  020000                  MWORD   <JMAP,J=2>                              ; 2
5289  005601' 000000  000040
5290  005602' 777777  777777                  -1
```

```
                                  ;#*********************************************************************
                                  ;* TEST 64 - IOP Function 0 - 40+2N Interrupt
                                  ;
                                  ; Description:  This test verifies that microcode can request an
                                  ;               interrupt with an IOP word specifying function 0
                                  ;               and this will cause a non-vectored  interrupt to
                                  ;               40+2N of the EPT.
                                  ;
                                  ; Procedure:    KL> Port Clear
                                  ;               KL> Load test ucode
                                  ;               KL> Set up PI system so that interrupts may occur
                                  ;               KL> Set CSR Register PIA bits 33-35 to 4
                                  ;               KL> Write CSR to set MPROC Run
                                  ;               KL> Process the interrupt and verify that a non-
                                  ;                   vectored interrupt occurred on PI level 4
                                  ;
                                  ;               Do for IOP words specifying:
                                  ;
                                  ;                AdrSpa    Func  Qual  Device  Addr   IOP Word
                                  ;                 EPT        0     0     0        0  000000,,000000
                                  ;                 EPT        0     1     0        0  004000,,000000
                                  ;                Exec Vir    0     0     0        0  100000,,000000
                                  ;                Exec Vir    0     1     0        0  104000,,000000
                                  ;                Physical    0     0     0        0  400000,,000000
                                  ;                Physical    0     1     0        0  404000,,000000
                                  ;                 EPT        0     0     0     1000  000000,,001000
                                  ;                 EPT        0     1     0     1000  004000,,001000
                                  ;                Exec Vir    0     0     0     1000  100000,,001000
                                  ;                Exec Vir    0     1     0     1000  104000,,001000
                                  ;                Physical    0     0     0     1000  400000,,001000
                                  ;                Physical    0     1     0     1000  404000,,001000
                                  ;
                                  ; Failure:      ---
                                  ;#*********************************************************************

                                  ; Test data

005603' 254 00 0 00 005611'       TSTE64: JRST    TG64                            ; go start test
005604' 400400  000064                    EBUS!NDMP!ZEBUS!64                      ; test mask
005605' 005703' 012445'                   T64M,,[ASCIZ ^IOP Function 0 - 40+2N Interrupt^]
005606' 012276' 012300'                   [EXP E14,MLAST!E7],,[EXP E4,MLAST!E1]
005607' 000000  006030'                   TSTE65                                  ; failure test table
005610' 777777  777777                    -1
```

```

                                        ; Start test

005611' 332 00 0 00 030037      TG64:   SKIPE   USER                            ; user mode?
                                        GO      [MOVE   TSTFLG                  ; yes - ensure this test does not
                                                 TLO    (TUSER)                 ;    get run (but only if not in
                                                 SKIPN  UDEBUG                  ;    debug mode)
                                                 MOVEM  TSTFLG
005612' 260 17 0 00 010511'                      RTN]
005613' 201 00 0 00 000000'             MOVEI   Z2                              ; get address of module start
005614' 260 17 0 00 005515*             GO      TRACE                           ; handle trace output
005615' 201 01 0 00 005703'             MOVEI   1,T64M                          ; set up microcode address
005616' 260 17 0 00 005517*             GO      TLOAD                           ; load/verify it
005617' 263 17 0 00 000000              RTN                                     ; failed - exit test

                                ; Initialization

005620' 400 15 0 00 000000      TL64:   SETZ    ERFLG,                          ; clear error flag
005621' 201 06 0 00 005635'             MOVEI   6,TS64                          ; get test table address
005622' 402 00 0 00 005555*             SETZM   TSTSUB                          ; initialize subtest number

                                ; Loop on test execute table entries

005623' 260 17 0 00 005337*     TA64:   GO      IPACLR                          ; clear port
005624' 260 17 0 00 000000*     TB64:   GO      IEXEC                           ; execute table entry
005625' 254 00 0 00 005634'              JRST   TX64                            ; end of test execute table
005626' 254 00 0 00 005624'              JRST   TB64                            ; keep looping after call
005627' 474 15 0 00 000000               SETO   ERFLG,                          ; error occurred

                                ; Handle error printouts and scope looping

005630' 027 00 0 00 005701'             SCOPER  MA64                            ; print error message
005631' 254 00 0 00 005620'             JRST    TL64                            ; loop on error
005632' 254 00 0 00 005634'             JRST    TX64                            ; altmode exit
005633' 322 15 0 00 005623'             JUMPE   ERFLG,TA64                      ; do next test execute table entry

                                ; End of test

005634' 263 17 0 00 000000      TX64:   RTN                                     ; return
```

```

                                        ; Interrupt Test Execute Table, as:  (CMD,parameters)

005635' 300000  005652'                 TS64:   ITABLE  (ICALL,T64ADI)          ; initialize start address
005636' 140004  005650'                         ITABLE  (ISETID,4,TS64Z)        ; set up default inc/dec data
005637' 040002  005650'                         ITABLE  (ISETEX,2,TS64Z)        ; set up default examine data
005640' 100002  005650'                         ITABLE  (ISETDE,2,TS64Z)        ; set up default deposit data
005641' 340000  005654'                 TS64A:  ITABLE  (ICALLC,T64ADD)         ; increment test address
005642' 200000  000004                          ITABLE  (ISETIN,0,4)            ; set up interrupt
005643' 500000  777777                          ITABLE  (ISTART,777777)         ; start at default address
005644' 400000  000010                          ITABLE  (IWAIT,10)              ; wait for 10 ms.
005645' 540000  000000                          ITABLE  (IEXIT)                 ; exit test if any error yet
005646' 440000  005641'                         ITABLE  (IJRST,TS64A)           ; loop till done
005647' 000000  000000                          ITABLE  (ILAST)                 ; end of table

005650' 000000  000000                  TS64Z:  0
005651' 000000  000000                          0

                                        ; Start address handling routines

005652' 476 00 0 00 013667'             T64ADI: SETOM   T64ADR#                 ; initialize start address
005653' 263 17 0 00 000000                      RTN                             ; return

005654' 350 01 0 00 013667'             T64ADD: AOS     1,T64ADR                ; increment address
                                                MOVE    [000000,,000000         ; get IOP function word
                                                         004000,,000000
                                                         100000,,000000
                                                         104000,,000000
                                                         400000,,000000
                                                         404000,,000000
                                                         000000,,001000
                                                         004000,,001000
                                                         100000,,001000
                                                         104000,,001000
                                                         400000,,001000
005655' 200 00 0 01 012454'                              404000,,001000](1)
005656' 202 00 0 00 000000*                     MOVEM   IIOPF                   ; save it
005657' 200 01 0 01 005664'                     MOVE    1,T64ADT(1)             ; use dispatch table
005660' 202 01 0 00 000000*                     MOVEM   1,IADDR                 ; set up start address
005661' 302 01 0 00 000777                      CAIE    1,777                   ; done yet?
005662' 350 00 0 17 000000                      AOS     (P)                     ; no - set up ok return
005663' 263 17 0 00 000000                      RTN                             ; return
```

```

                                        ; Start address table

        005664' 000000  000000          T64ADT: 0
        005665' 000000  000001                  1
        005666' 000000  000002                  2
        005667' 000000  000003                  3
        005670' 000000  000004                  4
        005671' 000000  000005                  5
        005672' 000000  000006                  6
        005673' 000000  000012                  12
        005674' 000000  000016                  16
        005675' 000000  000022                  22
        005676' 000000  000026                  26
        005677' 000000  000032                  32
        005700' 000000  000777                  777

                                        ; Error messages

        005701' 140000  012470'         MA64:   MSG!TXNOT![ASCIZ /IOP error detected/]
        005702' 000000000000#                   LAST!CALL!TXALL!IIPNT           ; print test data

                                        ; Microcode:

                                        ;       IOP = 000000,,000000    EPT / NVInt / Qual=0 / Addr=0
                                        ;       IOP = 004000,,000000    EPT / NVInt / Qual=1 / Addr=0
                                        ;       IOP = 100000,,000000    Exec Vir / NVInt / Qual=0 / Addr=0
                                        ;       IOP = 104000,,000000    Exec Vir / NVInt / Qual=1 / Addr=0
                                        ;       IOP = 400000,,000000    Physical / NVInt / Qual=0 / Addr=0
                                        ;       IOP = 404000,,000000    Physical / NVInt / Qual=1 / Addr=0

        005703' 000007  000000          T64M:   MWORD   <ADDR=0,JMAP,J=700,SD0,OR,D=2,SKCN,MGC=0>
        005704' 732000  240040
        005705' 000107  000004                  MWORD   <JMAP,J=700,SD0,OR,D=2,SKCN,MGC=4>      ; 1
        005706' 732000  240040
        005707' 000207  000100                  MWORD   <JMAP,J=700,SD0,OR,D=2,SKCN,MGC=100>    ; 2
        005710' 732000  240040
        005711' 000307  000104                  MWORD   <JMAP,J=700,SD0,OR,D=2,SKCN,MGC=104>    ; 3
        005712' 732000  240040
        005713' 000407  000400                  MWORD   <JMAP,J=700,SD0,OR,D=2,SKCN,MGC=400>    ; 4
        005714' 732000  240040
        005715' 000507  000404                  MWORD   <JMAP,J=700,SD0,OR,D=2,SKCN,MGC=404>    ; 5
        005716' 732000  240040

                                        ;       IOP = 000000,,001000    EPT / NVInt / Qual=0 / Addr=1000

        005717' 000600  070000                  MWORD   <JMAP,J=7,SOA,AND,D=2>                  ; 6
        005720' 442000  000040
        005721' 000700  140000                  MWORD   <LDCT,J=14,SAB,PLUS,D=2,CRY>            ; 7
        005722' 102000  000700
        005723' 001010  000000                  MWORD   <CJS,J=1000>                            ; 10
        005724' 000000  000020
        005725' 001107  010000                  MWORD   <JMAP,J=701,SD0,OR,D=2,SKCN,MGC=0>      ; 11
        005726' 732000  240040

```

```
                                        ;       IOP = 004000,,001000    EPT / NVInt / Qual=1 / Addr=1000

005727' 001200  130000                          MWORD   <JMAP,J=13,SOA,AND,D=2>                 ; 12
005730' 442000  000040
005731' 001300  140000                          MWORD   <LDCT,J=14,SAB,PLUS,D=2,CRY>            ; 13
005732' 102000  000700
005733' 001410  000000                          MWORD   <CJS,J=1000>                            ; 14
005734' 000000  000020
005735' 001507  010004                          MWORD   <JMAP,J=701,SDO,OR,D=2,SKCN,MGC=4>      ; 15
005736' 732000  240040

                                        ;       IOP = 100000,,001000    Exec Vir / NVInt / Qual=0 / Addr=1000

005737' 001600  170000                          MWORD   <JMAP,J=17,SOA,AND,D=2>                 ; 16
005740' 442000  000040
005741' 001700  140000                          MWORD   <LDCT,J=14,SAB,PLUS,D=2,CRY>            ; 17
005742' 102000  000700
005743' 002010  000000                          MWORD   <CJS,J=1000>                            ; 20
005744' 000000  000020
005745' 002107  010100                          MWORD   <JMAP,J=701,SDO,OR,D=2,SKCN,MGC=100>    ; 21
005746' 732000  240040

                                        ;       IOP = 104000,,001000    Exec Vir / NVInt / Qual=1 / Addr=1000

005747' 002200  230000                          MWORD   <JMAP,J=23,SOA,AND,D=2>                 ; 22
005750' 442000  000040
005751' 002300  140000                          MWORD   <LDCT,J=14,SAB,PLUS,D=2,CRY>            ; 23
005752' 102000  000700
005753' 002410  000000                          MWORD   <CJS,J=1000>                            ; 24
005754' 000000  000020
005755' 002507  010104                          MWORD   <JMAP,J=701,SDO,OR,D=2,SKCN,MGC=104>    ; 25
005756' 732000  240040

                                        ;       IOP = 400000,,001000    Physical / NVInt / Qual=0 / Addr=1000

005757' 002600  270000                          MWORD   <JMAP,J=27,SOA,AND,D=2>                 ; 26
005760' 442000  000040
005761' 002700  140000                          MWORD   <LDCT,J=14,SAB,PLUS,D=2,CRY>            ; 27
005762' 102000  000700
005763' 003010  000000                          MWORD   <CJS,J=1000>                            ; 30
005764' 000000  000020
005765' 003107  010400                          MWORD   <JMAP,J=701,SDO,OR,D=2,SKCN,MGC=400>    ; 31
005766' 732000  240040

                                        ;       IOP = 404000,,001000    Physical / NVInt / Qual=1 / Addr=1000

005767' 003200  330000                          MWORD   <JMAP,J=33,SOA,AND,D=2>                 ; 32
005770' 442000  000040
005771' 003300  140000                          MWORD   <LDCT,J=14,SAB,PLUS,D=2,CRY>            ; 33
005772' 102000  000700
005773' 003410  000000                          MWORD   <CJS,J=1000>                            ; 34
005774' 000000  000020
005775' 003507  010404                          MWORD   <JMAP,J=701,SDO,OR,D=2,SKCN,MGC=404>    ; 35
005776' 732000  240040

```

```
                                        ;       Subroutine to shift the right justified bit field in R0 to
                                        ;       bits 1-10, all other bits zero.  Then to place in R5 and
                                        ;       in EBUF to prepare for IOP request.  Then request interrupt
                                        ;       and halt.

005777' 070000  000000                          MWORD   <ADDR=700,CONT,SOA,AND,B=5,D=2>         ; 700
006000' 442002  400340
006001' 070100  320000                          MWORD   <LDCT,J=32,D=1>                         ; 701
006002' 001000  000300
006003' 070207  020000                          MWORD   <RPCT,J=702,SOA,OR,D=7>                 ; 702
006004' 437000  000220
006005' 070300  002004                          MWORD   <CONT,SAB,OR,D=3,B=5,OENA,SELE,MGC=4>   ; 703
006006' 133002  405340
006007' 070400  000002                          MWORD   <CONT,SELE,MGC=2>                       ; 704
006010' 000000  005340
006011' 070507  050000                          MWORD   <JMAP,J=705>                            ; 705
006012' 000000  000040

                                        ;       Subroutine to shift a bit over in R0, insert into R5, then
                                        ;       zero R0.

006013' 100000  000000                          MWORD   <ADDR=1000,CONT,SOA,AND,D=2,B=5>        ; 1000
006014' 442002  400340
006015' 100110  010000                          MWORD   <RPCT,J=1001,SOA,OR,D=7>                ; 1001
006016' 437000  000220
006017' 100200  000000                          MWORD   <CONT,SAB,OR,D=2,B=5>                   ; 1002
006020' 132002  400340
006021' 100300  000000                          MWORD   <CRTN,SOA,AND,D=2>                      ; 1003
006022' 442000  000240

                                        ;       Subroutine to read R5

006023' 110011  012004                          MWORD   <ADDR=1100,JMAP,J=1101,SOA,A=5,OR,D=1,OENA,SELE,MGC=4>
006024' 431050  005040
006025' 110111  010000                          MWORD   <JMAP,J=1101>                           ; 1101
006026' 000000  000040
006027' 777777  777777                          -1
```

```

                                      ;#*********************************************************************
                                      ;* TEST 65 - IOP Function 1 - 40+2N Interrupt
                                      ;
                                      ; Description:  This test verifies that microcode can request an
                                      ;               interrupt with an IOP word specifying function 1
                                      ;               and this will cause a non-vectored  interrupt to
                                      ;               40+2N of the EPT.
                                      ;
                                      ; Procedure:    KL> Port Clear
                                      ;               KL> Load test ucode
                                      ;               KL> Set up PI system so that interrupts may occur
                                      ;               KL> Set CSR Register PIA bits 33-35 to 4
                                      ;               KL> Write CSR to set MPROC Run
                                      ;               KL> Process the interrupt and verify that a non-
                                      ;                   vectored interrupt occurred on PI level 4
                                      ;
                                      ;               Do for IOP words specifying:
                                      ;
                                      ;                AdrSpa     Func  Qual  Device  Addr   IOP Word
                                      ;                 EPT        1     0      0       0   010000,,000000
                                      ;                 EPT        1     1      0       0   014000,,000000
                                      ;                Exec Vir    1     0      0       0   110000,,000000
                                      ;                Exec Vir    1     1      0       0   114000,,000000
                                      ;                Physical    1     0      0       0   410000,,000000
                                      ;                Physical    1     1      0       0   414000,,000000
                                      ;                 EPT        1     0      0    1000   010000,,001000
                                      ;                 EPT        1     1      0    1000   014000,,001000
                                      ;                Exec Vir    1     0      0    1000   110000,,001000
                                      ;                Exec Vir    1     1      0    1000   114000,,001000
                                      ;                Physical    1     0      0    1000   410000,,001000
                                      ;                Physical    1     1      0    1000   414000,,001000
                                      ;
                                      ; Failure:      ---
                                      ;#*********************************************************************

                                      ; Test data

006030' 254 00 0 00 006035'           TSTE65: JRST    TG65                    ; go start test
006031' 400400  000065                        EBUS!NDMP!ZEBUS!65              ; test mask
006032' 006127' 012474'                       T65M,,[ASCIZ ^IOP Function 1 - 40+2N Interrupt^]
006033' 012276' 012300'                       [EXP E:4,MLAST!E7],,[EXP E4,MLAST!E1]
006034' 777777  777777                        -1

                                      ; Start test

006035' 332 00 0 00 030037            TG65:   SKIPE   USER                    ; user mode?
                                              GO      [MOVE   TSTFLG          ; yes - ensure this test does not
                                                       TLO    (TUSER)         ;    get run (but only if not in
                                                       SKIPN  UDEBUG          ;    debug mode)
                                                       MOVEM  TSTFLG
006036' 260 17 0 00 010511'                            RTN]
```

```
006037' 201 00 0 00 000000'                  MOVEI   Z2                      ; get address of module start
006040' 260 17 0 00 005614*                  GO      TRACE                   ; handle trace output
006041' 201 01 0 00 006127'                  MOVEI   1,T65M                  ; set up microcode address
006042' 260 17 0 00 005616*                  GO      TLOAD                   ; load/verify it
006043' 263 17 0 00 000000                   RTN                             ; failed - exit test

                                     ; Initialization

006044' 400 15 0 00 000000           TL65:   SETZ    ERFLG,                  ; clear error flag
006045' 201 06 0 00 006061'                  MOVEI   6,TS65                  ; get test table address
006046' 402 00 0 00 005622*                  SETZM   TSTSUB                  ; initialize subtest number

                                     ; Loop on test execute table entries

006047' 260 17 0 00 005623*          TA65:   GO      IPACLR                  ; clear port
006050' 260 17 0 00 005624*          TB65:   GO      IEXEC                   ; execute table entry
006051' 254 00 0 00 006060'                   JRST   TX65                    ; end of test execute table
006052' 254 00 0 00 006050'                   JRST   TB65                    ; keep looping after call
006053' 474 15 0 00 000000                    SETO   ERFLG,                  ; error occurred

                                     ; Handle error printouts and scope looping

006054' 027 00 0 00 006125'                  SCOPER  MA65                    ; print error message
006055' 254 00 0 00 006044'                  JRST    TL65                    ; loop on error
006056' 254 00 0 00 006060'                  JRST    TX65                    ; altmode exit
006057' 322 15 0 00 006047'                  JUMPE   ERFLG,TA65              ; do next test execute table entry

                                     ; End of test

006060' 263 17 0 00 000000           TX65:   RTN                             ; return

                                     ; Interrupt Test Execute Table, as:  (CMD,parameters)

006061' 300000  006076'              TS65:   ITABLE  (ICALL,T65ADI)          ; initialize start address
006062' 140004  006074'                      ITABLE  (ISETID,4,TS65Z)        ; set up default inc/dec data
006063' 040002  006074'                      ITABLE  (ISETEX,2,TS65Z)        ; set up default examine data
006064' 100002  006074'                      ITABLE  (ISETDE,2,TS65Z)        ; set up default deposit data
006065' 340000  006100'              TS65A:  ITABLE  (ICALLC,T65ADD)         ; increment test address
006066' 200000  000004                       ITABLE  (ISETIN,0,4)            ; set up interrupt
006067' 500000  777777                       ITABLE  (ISTART,777777)         ; start at default address
006070' 400000  000010                       ITABLE  (IWAIT,10)              ; wait for 10 ms.
006071' 540000  000000                       ITABLE  (IEXIT)                 ; exit test if any error yet
006072' 440000  006065'                      ITABLE  (IJRST,TS65A)           ; loop till done
006073' 000000  000000                       ITABLE  (ILAST)                 ; end of table

006074' 000000  000000               TS65Z:  0
006075' 000000  000000                       0
```

```

                                       ; Start address handling routines

006076' 476 00 0 00 013670'            T65ADI: SETOM   T65ADR#                 ; initialize start address
006077' 263 17 0 00 000000                     RTN                             ; return

006100' 350 01 0 00 013670'            T65ADD: AOS     1,T65ADR                ; increment address
                                               MOVE    [010000,,000000         ; get IOP function word
                                                        014000,,000000
                                                        110000,,000000
                                                        114000,,000000
                                                        410000,,000000
                                                        414000,,000000
                                                        010000,,001000
                                                        014000,,001000
                                                        110000,,001000
                                                        114000,,001000
                                                        410000,,001000
006101' 200 00 0 01 012503'                             414000,,001000](1)
006102' 202 00 0 00 005656*                    MOVEM   IIOPF                   ; save it
006103' 200 01 0 01 006110'                    MOVE    1,T65ADT(1)             ; use dispatch table
006104' 202 01 0 00 005660*                    MOVEM   1,IADDR                 ; set up start address
006105' 302 01 0 00 000777                     CAIE    1,777                   ; done yet?
006106' 350 00 0 17 000000                     AOS     (P)                     ; no - set up ok return
006107' 263 17 0 00 000000                     RTN                             ; return

                                       ; Start address table

006110' 000000  000000                 T65ADT: 0
006111' 000000  000001                         1
006112' 000000  000002                         2
006113' 000000  000003                         3
006114' 000000  000004                         4
006115' 000000  000005                         5
006116' 000000  000006                         6
006117' 000000  000012                         12
006120' 000000  000016                         16
006121' 000000  000022                         22
006122' 000000  000026                         26
006123' 000000  000032                         32
006124' 000000  000777                         777

                                       ; Error messages

006125' 140000  012470'                MA65:   MSG!TXNOT![ASCIZ /IOP error detected/]
006126' 000000000000#                          LAST!CALL!TXALL!IIPNT           ; print test data
```

```

                                ; Microcode:

                                ;       IOP = 010000,,000000    EPT / NVInt / Qual=0 / Addr=0
                                ;       IOP = 014000,,000000    EPT / NVInt / Qual=1 / Addr=0
                                ;       IOP = 110000,,000000    Exec Vir / NVInt / Qual=0 / Addr=0
                                ;       IOP = 114000,,000000    Exec Vir / NVInt / Qual=1 / Addr=0
                                ;       IOP = 410000,,000000    Physical / NVInt / Qual=0 / Addr=0
                                ;       IOP = 414000,,000000    Physical / NVInt / Qual=1 / Addr=0

006127' 000007  000010          T65M:   MWORD   <ADDR=0,JMAP,J=700,SDO,OR,D=2,SKCN,MGC=10>
006130' 732000  240040
006131' 000107  000014                  MWORD   <JMAP,J=700,SDO,OR,D=2,SKCN,MGC=14>     ; 1
006132' 732000  240040
006133' 000207  000110                  MWORD   <JMAP,J=700,SDO,OR,D=2,SKCN,MGC=110>    ; 2
006134' 732000  240040
006135' 000307  000114                  MWORD   <JMAP,J=700,SDO,OR,D=2,SKCN,MGC=114>    ; 3
006136' 732000  240040
006137' 000407  000410                  MWORD   <JMAP,J=700,SDO,OR,D=2,SKCN,MGC=410>    ; 4
006140' 732000  240040
006141' 000507  000414                  MWORD   <JMAP,J=700,SDO,OR,D=2,SKCN,MGC=414>    ; 5
006142' 732000  240040

                                ;       IOP = 010000,,001000    EPT / NVInt / Qual=0 / Addr=1000

006143' 000600  070000                  MWORD   <JMAP,J=7,SOA,AND,D=2>                  ; 6
006144' 442000  000040
006145' 000700  140000                  MWORD   <LDCT,J=14,SAB,PLUS,D=2,CRY>            ; 7
006146' 102000  000700
006147' 001010  000000                  MWORD   <CJS,J=1000>                            ; 10
006150' 000000  000020
006151' 001107  010010                  MWORD   <JMAP,J=701,SDO,OR,D=2,SKCN,MGC=10>     ; 11
006152' 732000  240040

                                ;       IOP = 014000,,001000    EPT / NVInt / Qual=1 / Addr=1000

006153' 001200  130000                  MWORD   <JMAP,J=13,SOA,AND,D=2>                 ; 12
006154' 442000  000040
006155' 001300  140000                  MWORD   <LDCT,J=14,SAB,PLUS,D=2,CRY>            ; 13
006156' 102000  000700
006157' 001410  000000                  MWORD   <CJS,J=1000>                            ; 14
006160' 000000  000020
006161' 001507  010014                  MWORD   <JMAP,J=701,SDO,OR,D=2,SKCN,MGC=14>     ; 15
006162' 732000  240040

                                ;       IOP = 110000,,001000    Exec Vir / NVInt / Qual=0 / Addr=1000

006163' 001600  170000                  MWORD   <JMAP,J=17,SOA,AND,D=2>                 ; 16
006164' 442000  000040
006165' 001700  140000                  MWORD   <LDCT,J=14,SAB,PLUS,D=2,CRY>            ; 17
006166' 102000  000700
006167' 002010  000000                  MWORD   <CJS,J=1000>                            ; 20
006170' 000000  000020
006171' 002107  010110                  MWORD   <JMAP,J=701,SDO,OR,D=2,SKCN,MGC=110>    ; 21
006172' 732000  240040
```

```
5763
5764                                  ;       IOP = 114000,,001000    Exec Vir / NVInt / Qual=1 / Addr=1000
5765
5766  006173' 002200  230000                  MWORD   <JMAP,J=23,SOA,AND,D=2>                       ; 22
5767  006174' 442000  000040
5768  006175' 002300  140000                  MWORD   <LDCT,J=14,SAB,PLUS,D=2,CRY>                  ; 23
5769  006176' 102000  000700
5770  006177' 002410  000000                  MWORD   <CJS,J=1000>                                  ; 24
5771  006200' 000000  000020
5772  006201' 002507  010114                  MWORD   <JMAP,J=701,SDO,OR,D=2,SKCN,MGC=114>          ; 25
5773  006202' 732000  240040
5774
5775                                  ;       IOP = 410000,,001000    Physical / NVInt / Qual=0 / Addr=1000
5776
5777  006203' 002600  270000                  MWORD   <JMAP,J=27,SOA,AND,D=2>                       ; 26
5778  006204' 442000  000040
5779  006205' 002700  140000                  MWORD   <LDCT,J=14,SAB,PLUS,D=2,CRY>                  ; 27
5780  006206' 102000  000700
5781  006207' 003010  000000                  MWORD   <CJS,J=1000>                                  ; 30
5782  006210' 000000  000020
5783  006211' 003107  010410                  MWORD   <JMAP,J=701,SDO,OR,D=2,SKCN,MGC=410>          ; 31
5784  006212' 732000  240040
5785
5786                                  ;       IOP = 414000,,001000    Physical / NVInt / Qual=1 / Addr=1000
5787
5788  006213' 003200  330000                  MWORD   <JMAP,J=33,SOA,AND,D=2>                       ; 32
5789  006214' 442000  000040
5790  006215' 003300  140000                  MWORD   <LDCT,J=14,SAB,PLUS,D=2,CRY>                  ; 33
5791  006216' 102000  000700
5792  006217' 003410  000000                  MWORD   <CJS,J=1000>                                  ; 34
5793  006220' 000000  000020
5794  006221' 003507  010414                  MWORD   <JMAP,J=701,SDO,OR,D=2,SKCN,MGC=414>          ; 35
5795  006222' 732000  240040
5796
5797                                  ;       Subroutine to shift the right justified bit field in R0 to
5798                                  ;       bits 1-10, all other bits zero.  Then to place in R5 and
5799                                  ;       in EBUF to prepare for IOP request.  Then request interrupt
5800                                  ;       and halt.
5801
5802  006223' 070000  000000                  MWORD   <ADDR=700,CONT,SOA,AND,B=5,D=2>               ; 700
5803  006224' 442002  400340
5804  006225' 070100  320000                  MWORD   <LDCT,J=32,D=1>                               ; 701
5805  006226' 001000  000300
5806  006227' 070207  020000                  MWORD   <RPCT,J=702,SOA,OR,D=7>                       ; 702
5807  006230' 437000  000220
5808  006231' 070300  002004                  MWORD   <CONT,SAB,OR,D=3,B=5,OENA,SELE,MGC=4>         ; 703
5809  006232' 133002  405340
5810  006233' 070400  000002                  MWORD   <CONT,SELE,MGC=2>                             ; 704
5811  006234' 000000  005340
5812  006235' 070507  050000                  MWORD   <JMAP,J=705>                                  ; 705
5813  006236' 000000  000040
5814
5815                                  ;       Subroutine to shift a bit over in R0, insert into R5, then
5816                                  ;       zero R0.
5817
```

```
  5818  006237' 100000  000000                    MWORD   <ADDR=1000,CONT,SOA,AND,D=2,B=5>           ; 1000
  5819  006240' 442002  400340
  5820  006241' 100110  010000                    MWORD   <RPCT,J=1001,SOA,OR,D=7>                   ; 1001
  5821  006242' 437000  000220
  5822  006243' 100200  000000                    MWORD   <CONT,SAB,OR,D=2,B=5>                      ; 1002
  5823  006244' 132002  400340
  5824  006245' 100300  000000                    MWORD   <CRTN,SOA,AND,D=2>                         ; 1003
  5825  006246' 442000  000240
  5826
  5827                                  ;         Subroutine to read R5
  5828
  5829  006247' 110011  012004                    MWORD   <ADDR=1100,JMAP,J=1101,SOA,A=5,OR,D=1,OENA,SELE,MGC=4>
  5830  006250' 431050  005040
  5831  006251' 110111  010000                    MWORD   <JMAP,J=1101>                              ; 1101
  5832  006252' 000000  000040
  5833  006253' 777777  777777                    -1
```

```


                                    ;#*********************************************************************
                                    ;* TEST 66 - x
                                    ;#*********************************************************************

                                    ; Test data

006254' 263 17 0 00 000000          TSTE66: RTN                                     ; exit
006255' 000000  000066                      ZEBUS!66                                ; test mask
006256' 000000  012142'                     0,,[ASCIZ ^x^]
006257' 000000  000000                      0,,0
006260' 777777  777777                      -1


                                    ;#*********************************************************************
                                    ;* TEST 67 - x
                                    ;#*********************************************************************

                                    ; Test data

006261' 263 17 0 00 000000          TSTE67: RTN                                     ; exit
006262' 000000  000067                      ZEBUS!67                                ; test mask
006263' 000000  012142'                     0,,[ASCIZ ^x^]
006264' 000000  000000                      0,,0
006265' 777777  777777                      -1
```

```

                                          ;#*********************************************************************
                                          ;* TEST 70 - IOP Function 4 - Examine
                                          ;
                                          ; Description:   This test verifies that microcode can request an
                                          ;                interrupt with an IOP word specifying function 4
                                          ;                and this will cause the  location  specified  by
                                          ;                the address  portion  of the function word to be
                                          ;                placed on the EBUS for reading by the port.
                                          ;
                                          ; Procedure:     KL> Port Clear
                                          ;                KL> Load test ucode
                                          ;                KL> Set up PI system so that interrupts may occur
                                          ;                KL> Set CSR Register PIA bits 33-35 to 0
                                          ;                KL> Write CSR to set MPROC Run
                                          ;                KL> Stop the port, then read the EBUF and verify
                                          ;                    that the data pattern specified is now in
                                          ;                    the EBUF.
                                          ;
                                          ;                Data pattern to use:  525252,,777777
                                          ;
                                          ;                Do for IOP words specifying:
                                          ;
                                          ;                 AdrSpa     Func   Qual  Device  Addr   IOP Word
                                          ;                 Physical    4      0      0     BUFF  440000,,BUFF
                                          ;                   EPT       4      0      0      100  040000,,000100
                                          ;                 Exec Vir    4      0      0      500  140000,,000500
                                          ;                 Physical    4      0      0      600  440000,,000600
                                          ;
                                          ; Failure:       ---
                                          ;#*********************************************************************

                                          ; Test data

 006266' 254 00 0 00 006275'      TSTE70: JRST    TG70                          ; go start test
 006267' 400400  000070                   EBUS!NDMP!ZEBUS!70                    ; test mask
 006270' 006425' 012517'                  T70M,,[ASCIZ ^IOP Function 4 - Examine^]
 006271' 012276' 012300'                  [EXP E14,MLAST!E7],,[EXP E4,MLAST!E1]
 006272' 000000  006522'                  TSTE71                                ; failure test table
 006273' 000000  006764'                  TSTE72                                ;   ...
 006274' 777777  777777                   -1

                                          ; Start test

 006275' 201 00 0 00 000000'      TG70:   MOVEI   Z2                            ; get address of module start
 006276' 260 17 0 00 006040*              GO      TRACE                         ; handle trace output
 006277' 201 01 0 00 006425'              MOVEI   1,T70M                        ; set up microcode address
 006300' 260 17 0 00 006042*              GO      TLOAD                         ; load/verify it
 006301' 263 17 0 00 000000               RTN                                   ; failed - exit test
```

```

                                        ; Initialization

006302' 201 01 0 00 000001                      MOVEI   1,1                             ; set to 'read'
006303' 201 02 0 00 000010                      MOVEI   2,10                            ; set to 10 words
006304' 201 14 0 00 000010                      MOVEI   PAT,10                          ; set up data pattern
006305' 260 17 0 00 000000*                     GO      BUFGEN                          ; build CCW list (locks BUFF in core)

006306' 400 15 0 00 000000              TL70:   SETZ    ERFLG,                          ; clear error flag
006307' 201 06 0 00 006323'                     MOVEI   6,TS70                          ; get test table address
006310' 402 00 0 00 006046*                     SETZM   TSTSUB                          ; initialize subtest number

                                        ; Loop on test execute table entries

006311' 260 17 0 00 006047*             TA70:   GO      IPACLR                          ; clear port
006312' 260 17 0 00 006050*             TB70:   GO      IEXEC                           ; execute table entry
006313' 254 00 0 00 006322'                      JRST   TX70                            ; end of test execute table
006314' 254 00 0 00 006312'                      JRST   TB70                            ; keep looping after call
006315' 474 15 0 00 000000                       SETO   ERFLG,                          ; error occurred

                                        ; Handle error printouts and scope looping

006316' 027 00 0 00 006423'                     SCOPER  MA70                            ; print error message
006317' 254 00 0 00 006306'                     JRST    TL70                            ; loop on error
006320' 254 00 0 00 006322'                     JRST    TX70                            ; altmode exit
006321' 322 15 0 00 006311'                     JUMPE   ERFLG,TA70                      ; do next test execute table entry

                                        ; End of test

006322' 263 17 0 00 000000              TX70:   RTN                                     ; return

                                        ; Interrupt Test Execute Table, as:  (CMD,parameters)

006323' 140004  006342'                 TS70:   ITABLE  (ISETID,4,TS70Z)                ; set up increment/decrement data
006324' 100002  006342'                         ITABLE  (ISETDE,2,TS70Z)                ; set up default deposit data

006325' 300000  006371'                         ITABLE  (ICALL,TS70EX)                  ; set up test data
006326' 040002  006344'                         ITABLE  (ISETEX,2,TS70AA)               ; set up examine data
006327' 500000  000100                          ITABLE  (ISTART,100)                    ; start at 100
006330' 400000  000010                          ITABLE  (IWAIT,10)                      ; wait for 10 ms.
006331' 540000  000000                          ITABLE  (IEXIT)                         ; exit test if any error yet

006332' 340000  006346'                         ITABLE  (ICALLC,T70ADI)                 ; initialize start address
006333' 340000  006352'                 TS70A:  ITABLE  (ICALLC,T70ADD)                 ; increment test address
006334' 040002  006344'                         ITABLE  (ISETEX,2,TS70AA)               ; set up examine data
006335' 500000  777777                          ITABLE  (ISTART,777777)                 ; start at default address
006336' 400000  000010                          ITABLE  (IWAIT,10)                      ; wait for 10 ms.
006337' 540000  000000                          ITABLE  (IEXIT)                         ; exit test if any error yet
006340' 440000  006333'                         ITABLE  (IJRST,TS70A)                   ; loop till done
006341' 000000  000000                          ITABLE  (ILAST)                         ; end of table

```

```

006342' 000000  000000          TS70Z:  0
006343' 000000  000000                  0

006344' 000000  000000          TS70AA: 0
006345' 000000  000000                  0

                                ; Start address handling routines

006346' 476 00 0 00 013671'     T70ADI: SETOM   T70ADR#                 ; initialize start address
006347' 336 00 0 00 030037              SKIPN   USER                    ; done with test (user mode)?
006350' 350 00 0 17 000000              AOS     (P)                     ; no - set up ok return
006351' 263 17 0 00 000000              RTN                             ; return

006352' 350 01 0 00 013671'     T70ADD: AOS     1,T70ADR                ; increment address
                                        MOVE    [040000,,000100         ; get IOP function word
                                                 140000,,000500
006353' 200 00 0 01 012524'                      440000,,000600](1)
006354' 202 00 0 00 006102*             MOVEM   IIOPF                   ; save it
006355' 200 00 0 00 012527'             MOVE    [525252,,777777]        ; initial data
006356' 202 00 0 00 340100              MOVEM   340100                  ; save it location 100
006357' 202 00 0 00 340500              MOVEM   340500                  ; save in location 400
006360' 202 00 0 00 340600              MOVEM   340600                  ; save in location 600
006361' 202 00 0 00 006345'             MOVEM   TS70AA+1                ; save correct data
006362' 200 01 0 01 006417'             MOVE    1,T70ADT(1)             ; use dispatch table
006363' 202 01 0 00 006104*             MOVEM   1,IADDR                 ; set up start address
006364' 201 00 0 00 000010              MOVEI   10                      ; get start CSR data and save in SDATA
006365' 202 00 0 00 005530*             MOVEM   SDATA                   ;   (this ensures PI bits not set)
006366' 302 01 0 00 000777              CAIE    1,777                   ; done yet?
006367' 350 00 0 17 000000              AOS     (P)                     ; no - set up ok return
006370' 263 17 0 00 000000              RTN                             ; return

                                ; Initial examine routine

006371' 200 00 0 00 012530'     TS70EX: MOVE    [440000,,BUFF]          ; get IOP function word
006372' 202 00 0 00 006354*             MOVEM   IIOPF                   ; save it
006373' 200 00 0 00 012527'             MOVE    [525252,,777777]        ; initial data
006374' 202 00 0 00 005225*             MOVEM   BUFF                    ; save it
006375' 202 00 0 00 006345'             MOVEM   TS70AA+1                ; save correct
006376' 550 02 0 00 006372*             HRRZ    2,IIOPF                 ; get address to be examined
006377' 202 02 0 00 006344'             MOVEM   2,TS70AA                ; save it
006400' 336 00 0 00 005203*             SKIPN   UDEBUG                  ; debug mode?
006401' 257 02 0 02 000000              MAP     2,(2)                   ; no - map to physical
006402' 621 02 0 00 777000              TLZ     2,777000                ; mask off unused bits
006403' 201 01 0 00 000300              MOVEI   1,300                   ; get start address
006404' 242 01 0 00 000001              LSH     1,1                     ; convert to address for RAR
006405' 260 17 0 00 005172*             GO      LDRAR                   ; load RAR
006406' 201 01 0 00 000010              MOVEI   1,MPRUN                 ; get start data
006407' 260 17 0 00 005174*             GO      LDCSR                   ; start the port
```

```
006410' 200 01 0 00 012303'              MOVE    1,[DATAO 2]                ; build a DATAO
006411' 434 01 0 00 000016               IOR     1,MBCN                     ; include the device code
006412' 336 00 0 00 006400*              SKIPN   UDEBUG                     ; debug mode?
006413' 256 00 0 00 000001               XCT     1                          ; no - do the DATAO
006414' 400 01 0 00 000000               SETZ    1,                         ; clear CSR data
006415' 260 17 0 00 006407*              GO      LDCSR                      ; stop the port
006416' 263 17 0 00 000000               RTN                                ; return

                                 ; Start address table

006417' 000000  000000           T70ADT: 0
006420' 000000  000003                   3
006421' 000000  000006                   6
006422' 000000  000777                   777

                                 ; Error messages

006423' 140000  012470'          MA70:   MSG!TXNOT![ASCIZ /IOP error detected/]
006424' 000000000000#                    LAST!CALL!TXALL!IIPNT              ; print test data

                                 ; Microcode:

                                 ;       IOP = 040000,,000100    EPT / Examine / Qual=0 / Addr=100

006425' 000000  010000           T70M:   MWORD   <ADDR=0,JMAP,J=1>                        ; 0
006426' 000000  000040
006427' 000106  000100                   MWORD   <CJS,J=600,SD0,OR,D=2,SKCN,MGC=100>      ; 1
006430' 732000  240020
006431' 000207  010040                   MWORD   <JMAP,J=701,SD0,OR,D=2,SKCN,MGC=40>      ; 2
006432' 732000  240040

                                 ;       IOP = 140000,,000500    Exec Vir / Examine / Qual=0 / Addr=500

006433' 000300  040000                   MWORD   <JMAP,J=4>                               ; 3
006434' 000000  000040
006435' 000406  000500                   MWORD   <CJS,J=600,SD0,OR,D=2,SKCN,MGC=500>      ; 4
006436' 732000  240020
006437' 000507  010140                   MWORD   <JMAP,J=701,SD0,OR,D=2,SKCN,MGC=140>     ; 5
006440' 732000  240040

                                 ;       IOP = 440000,,000600    Physical / Examine / Qual=0 / Addr=600

006441' 000600  070000                   MWORD   <JMAP,J=7>                               ; 6
006442' 000000  000040
006443' 000706  000600                   MWORD   <CJS,J=600,SD0,OR,D=2,SKCN,MGC=600>      ; 7
006444' 732000  240020
006445' 001007  010440                   MWORD   <JMAP,J=701,SD0,OR,D=2,SKCN,MGC=440>     ; 10
006446' 732000  240040

                                 ;       IOP = 440000,,BUFF      Physical / Examine / Qual=0 / Addr=BUFF

006447' 010001  010000                   MWORD   <ADDR=100,JMAP,J=101>                    ; 100
006450' 000000  000040
006451' 010107  010440                   MWORD   <JMAP,J=701,SD0,OR,D=2,SKCN,MGC=440>     ; 101
006452' 732000  240040
```

```

                                ;       Routine to read the buffer address and save in R5

006453' 030003  010000                  MWORD   <ADDR=300,JMAP,J=301>                           ; 300
006454' 000000  000040
006455' 030103  030000                  MWORD   <CJP,J=303,CENA,CCER>                           ; 301
006456' 000400  100060
006457' 030203  010000                  MWORD   <JMAP,J=301>                                    ; 302
006460' 000000  000040
006461' 030300  000010                  MWORD   <CONT,SDO,OR,D=2,B=5,SELE,MGC=10>               ; 303
006462' 732002  405340
006463' 030403  040000                  MWORD   <JMAP,J=304,D=1>                                ; 304
006464' 001000  000040

                                ;       Subroutine to insert right 10 bits of R0 into R5 zeroing
                                ;       the rest of R5.

006465' 060000  310000                  MWORD   <ADDR=600,LDCT,J=31,SOA,OR,B=5,D=2>             ; 600
006466' 432002  400300
006467' 060106  010000                  MWORD   <RPCT,J=601,SOB,B=5,OR,D=5>                     ; 601
006470' 335002  400220
006471' 060200  000000                  MWORD   <CRTN>                                          ; 602
006472' 000000  000240

                                ;       Subroutine to shift the right justified bit field in R0 to
                                ;       bits 0-8, all other bits zero.  Then request interrupt and
                                ;       wait for CC EbusRqst.  Then put EBUS data into R5 and halt.

006473' 070000  000000                  MWORD   <ADDR=700,CONT,SOA,AND,B=5,D=2>                 ; 700
006474' 442002  400340
006475' 070100  320000                  MWORD   <LDCT,J=32,D=1>                                 ; 701
006476' 001000  000300
006477' 070207  020000                  MWORD   <RPCT,J=702,SOA,OR,D=7>                         ; 702
006500' 437000  000220
006501' 070300  002004                  MWORD   <CONT,SAB,OR,B=5,D=3,OENA,SELE,MGC=4>           ; 703
006502' 133002  405340
006503' 070400  000001                  MWORD   <CONT,SELE,MGC=1>                               ; 704 (Ex/Dep int)
006504' 000000  005340
006505' 070507  070000                  MWORD   <CJP,J=707,D=1,CENA,CCER>                       ; 705
006506' 001400  100060
006507' 070607  050000                  MWORD   <JMAP,J=705,D=1>                                ; 706
006510' 001000  000040
006511' 070700  000010                  MWORD   <CONT,SDO,OR,D=2,B=5,SELE,MGC=10>               ; 707
006512' 732002  405340
006513' 071007  100000                  MWORD   <JMAP,J=710>                                    ; 710
006514' 000000  000040

                                ;       Subroutine to read R5

006515' 110011  012004                  MWORD   <ADDR=1100,JMAP,J=1101,SOA,A=5,OR,D=1,OENA,SELE,MGC=4>
006516' 431050  005040
006517' 110111  010000                  MWORD   <JMAP,J=1101>                                   ; 1101
006520' 000000  000040
006521' 777777  777777                  -1

```

```
                                   ;#*************************************************************************
                                   ;* TEST 71 - IOP Function 5 - Deposit
                                   ;
                                   ; Description:  This test verifies that microcode can request an
                                   ;               interrupt with an IOP word specifying function 5
                                   ;               and this will cause the  location  specified  by
                                   ;               the address  portion  of the function word to be
                                   ;               deposited with the data the port  places  on  the
                                   ;               EBUS.
                                   ;
                                   ; Procedure:    KL> Port Clear
                                   ;               KL> Load test ucode
                                   ;               KL> Set up PI system so that interrupts may occur
                                   ;               KL> Set CSR Register PIA bits 33-35 to 0
                                   ;               KL> Write CSR to set MPROC Run
                                   ;               KL> Examine the selected location and verify that
                                   ;                   it contains the correct data.
                                   ;
                                   ;               Data pattern to use:  652100,,001525
                                   ;
                                   ;               Do for IOP words specifying:
                                   ;
                                   ;                AdrSpa    Func  Qual  Device  Addr   IOP Word
                                   ;                Physical   5     0      0     BUFF  450000,,BUFF
                                   ;                 EPT       5     0      0      100  050000,,000100
                                   ;                Exec Vir   5     0      0      500  150000,,000500
                                   ;                Physical   5     0      0      600  450000,,000600
                                   ;
                                   ; Failure:      ---
                                   ;#*************************************************************************

                                   ; Test data

006522' 254 00 0 00 006530'        TSTE71: JRST    TG71                    ; go start test
006523' 400400  000071                     EBUS!NDMP!ZEBUS!71              ; test mask
006524' 006661' 012531'                    T71M,,[ASCIZ ^IOP Function 5 - Deposit^]
006525' 012276' 012300'                    [EXP E14,MLAST!E7],,[EXP E4,MLAST!E1]
006526' 000000  006764'                    TSTE72                          ; failure test table
006527' 777777  777777                     -1

                                   ; Start test

006530' 201 00 0 00 000000'        TG71:   MOVEI   Z2                      ; get address of module start
006531' 260 17 0 00 006276*                GO      TRACE                   ; handle trace output
006532' 201 01 0 00 006661'                MOVEI   1,T71M                  ; set up microcode address
006533' 260 17 0 00 006300*                GO      TLOAD                   ; load/verify it
006534' 263 17 0 00 000000                 RTN                             ; failed - exit test
```

```

                                        ; Initialization

006535' 201 01 0 00 000001                      MOVEI   1,1                     ; set to 'read'
006536' 201 02 0 00 000010                      MOVEI   2,10                    ; set to 10 words
006537' 201 14 0 00 000010                      MOVEI   PAT,10                  ; set up data pattern
006540' 260 17 0 00 006305*                     GO      BUFGEN                  ; build CCW list (locks BUFF in core)

006541' 400 15 0 00 000000              TL71:   SETZ    ERFLG,                  ; clear error flag
006542' 201 06 0 00 006556'                     MOVEI   6,TS71                  ; get test table address
006543' 402 00 0 00 006310*                     SETZM   TSTSUB                  ; initialize subtest number

                                        ; Loop on test execute table entries

006544' 260 17 0 00 006311*             TA71:   GO      IPACLR                  ; clear port
006545' 260 17 0 00 006312*             TB71:   GO      IEXEC                   ; execute table entry
006546' 254 00 0 00 006555'                      JRST   TX71                    ; end of test execute table
006547' 254 00 0 00 006545'                      JRST   TB71                    ; keep looping after call
006550' 474 15 0 00 000000                       SETO   ERFLG,                  ; error occurred

                                        ; Handle error printouts and scope looping

006551' 027 00 0 00 006657'                     SCOPER  MA71                    ; print error message
006552' 254 00 0 00 006541'                     JRST    TL71                    ; loop on error
006553' 254 00 0 00 006555'                     JRST    TX71                    ; altmode exit
006554' 322 15 0 00 006544'                     JUMPE   ERFLG,TA71              ; do next test execute table entry

                                        ; End of test

006555' 263 17 0 00 000000              TX71:   RTN                             ; return

                                        ; Interrupt Test Execute Table, as:  (CMD,parameters)

006556' 140004  006575'                 TS71:   ITABLE  (ISETID,4,TS71Z)        ; set up increment/decrement data
006557' 040002  006575'                         ITABLE  (ISETEX,2,TS71Z)        ; set up default examine data

006560' 300000  006626'                         ITABLE  (ICALL,TS71DE)          ; set up test data
006561' 100002  006577'                         ITABLE  (ISETDE,2,TS71AA)       ; set up examine data
006562' 500000  000100                          ITABLE  (ISTART,100)            ; start at 100
006563' 400000  000010                          ITABLE  (IWAIT,10)              ; wait for 10 ms.
006564' 540000  000000                          ITABLE  (IEXIT)                 ; exit test if any error yet

006565' 340000  006601'                         ITABLE  (ICALLC,T71ADI)         ; initialize start address
006566' 340000  006605'                 TS71A:  ITABLE  (ICALLC,T71ADD)         ; increment test address
006567' 100003  006577'                         ITABLE  (ISETDE,3,TS71AA)       ; set up deposit data
006570' 500000  777777                          ITABLE  (ISTART,777777)         ; start at default address
006571' 400000  000010                          ITABLE  (IWAIT,10)              ; wait for 10 ms.
006572' 540000  000000                          ITABLE  (IEXIT)                 ; exit test if any error yet
006573' 440000  006566'                         ITABLE  (IJRST,TS71A)           ; loop till done
006574' 000000  000000                          ITABLE  (ILAST)                 ; end of table

```

```

006575' 000000  000000        TS71Z:  0
006576' 000000  000000                0

006577' 000000  000000        TS71AA: 0
006600' 000000  001234                1234

                              ; Start address handling routines

006601' 476 00 0 00 013672'   T71ADI: SETOM   T71ADR#                 ; initialize start address
006602' 336 00 0 00 030037            SKIPN   USER                    ; done with test (user mode)?
006603' 350 00 0 17 000000            AOS     (P)                     ; no - set up ok return
006604' 263 17 0 00 000000            RTN                             ; return

006605' 350 01 0 00 013672'   T71ADD: AOS     1,T71ADR                ; increment address
                                      MOVE    [050000,,000100         ; get IOP function word
                                               150000,,000500
006606' 200 00 0 01 012536'                    450000,,000600](1)
006607' 202 00 0 00 006376*           MOVEM   IIOPF                   ; save it
006610' 200 00 0 00 010744'           MOVE    [400000,,1]             ; initial data
006611' 202 00 0 00 340100            MOVEM   340100                  ; save it location 100
006612' 202 00 0 00 340500            MOVEM   340500                  ; save in location 500
006613' 202 00 0 00 340600            MOVEM   340600                  ; save in location 600
006614' 550 00 0 00 006607*           HRRZ    IIOPF                   ; get address to be deposited to
006615' 202 00 0 00 006577'           MOVEM   TS71AA                  ; save it
006616' 200 01 0 00 013672'           MOVE    1,T71ADR                ; get address offset
006617' 200 01 0 01 006653'           MOVE    1,T71ADT(1)             ; use dispatch table
006620' 202 01 0 00 006363*           MOVEM   1,IADDR                 ; set up start address
006621' 201 00 0 00 000010            MOVEI   10                      ; get start CSR data and save in SDATA
006622' 202 00 0 00 006365*           MOVEM   SDATA                   ;   (this ensures PI bits not set)
006623' 302 01 0 00 000777            CAIE    1,777                   ; done yet?
006624' 350 00 0 17 000000            AOS     (P)                     ; no - set up ok return
006625' 263 17 0 00 000000            RTN                             ; return

                              ; Initial examine routine

006626' 200 00 0 00 012541'   TS71DE: MOVE    [450000,,BUFF]          ; get IOP function word
006627' 202 00 0 00 006614*           MOVEM   IIOPF                   ; save it
006630' 200 00 0 00 010744'           MOVE    [400000,,1]             ; initial data
006631' 202 00 0 00 006374*           MOVEM   BUFF                    ; save it
006632' 550 02 0 00 006627*           HRRZ    2,IIOPF                 ; get address to be examined
006633' 202 02 0 00 006577'           MOVEM   2,TS71AA                ; save it
006634' 336 00 0 00 006412*           SKIPN   UDEBUG                  ; debug mode?
006635' 257 02 0 02 000000            MAP     2,(2)                   ; no - map to physical
006636' 621 02 0 00 777000            TLZ     2,777000                ; mask off unused bits
006637' 201 01 0 00 000300            MOVEI   1,300                   ; get start address
006640' 242 01 0 00 000001            LSH     1,1                     ; convert to address for RAR
006641' 260 17 0 00 006405*           GO      LDRAR                   ; load RAR
006642' 201 01 0 00 000010            MOVEI   1,MPRUN                 ; get start data
006643' 260 17 0 00 006415*           GO      LDCSR                   ; start the port
```

```
006644' 200 01 0 00 012303'           MOVE    1,[DATAO 2]              ; build a DATAO
006645' 434 01 0 00 000016            IOR     1,MBCN                   ; include the device code
006646' 336 00 0 00 006634*           SKIPN   UDEBUG                   ; debug mode?
006647' 256 00 0 00 000001            XCT     1                        ; no - do the DATAO
006650' 400 01 0 00 000000            SETZ    1,                       ; clear CSR data
006651' 260 17 0 00 006643*           GO      LDCSR                    ; stop the port
006652' 263 17 0 00 000000            RTN                              ; return

                              ; Start address table

006653' 000000  000000        T71ADT: 0
006654' 000000  000003                3
006655' 000000  000006                6
006656' 000000  000777                777

                              ; Error messages

006657' 140000  012470'       MA71:   MSG!TXNOT![ASCIZ /IOP error detected/]
006660' 000000000000#                 LAST!CALL!TXALL!IIPNT            ; print test data

                              ; Microcode:

                              ;       IOP = 050000,,000100    EPT / Deposit / Qual = 0 / Addr= 100

006661' 000000  010000        T71M:   MWORD   <ADDR=0,JMAP,J=1>                   ; 0
006662' 000000  000040
006663' 000106  000100                MWORD   <CJS,J=600,SDO,OR,D=2,SKCN,MGC=100>     ; 1
006664' 732000  240020
006665' 000207  010050                MWORD   <JMAP,J=701,SDO,OR,D=2,SKCN,MGC=50>     ; 2
006666' 732000  240040

                              ;       IOP = 150000,,000500    Exec Vir / Deposit / Qual = 0 / Addr=500

006667' 000300  040000                MWORD   <JMAP,J=4>                          ; 3
006670' 000000  000040
006671' 000406  000500                MWORD   <CJS,J=[illegible]0,SDO,OR,D=2,SKCN,MGC=500>     ; 4
006672' 732000  240020
006673' 000507  010150                MWORD   <JMAP,J=701,SDO,OR,D=2,SKCN,MGC=150>    ; 5
006674' 732000  240040

                              ;       IOP = 450000,,000600    Physical / Deposit / Qual = 0 / Addr=600

006675' 000600  070000                MWORD   <JMAP,J=7>                          ; 6
006676' 000000  000040
006677' 000706  000600                MWORD   <CJS,J=600,SDO,OR,D=2,SKCN,MGC=600>     ; 7
006700' 732000  240020
006701' 001007  010450                MWORD   <JMAP,J=701,SDO,OR,D=2,SKCN,MGC=450>    ; 10
006702' 732000  240040

                              ;       IOP = 450000,,BUFF      Physical / Deposit / Qual=0 / Addr=BUFF

006703' 010001  010000                MWORD   <ADDR=100,JMAP,J=101>               ; 100
006704' 000000  000040
006705' 010107  010450                MWORD   <JMAP,J=701,SDO,OR,D=2,SKCN,MGC=450>    ; 101
006706' 732000  240040
```

```

                                        ;       Routine to read the buffer address and save in R5

006707' 030003  010000                          MWORD   <ADDR=300,JMAP,J=301>                           ; 300
006710' 000000  000040
006711' 030103  030000                          MWORD   <CJP,J=303,CENA,CCER>                           ; 301
006712' 000400  100060
006713' 030203  010000                          MWORD   <JMAP,J=301>                                    ; 302
006714' 000000  000040
006715' 030300  000010                          MWORD   <CONT,SDO,OR,D=2,B=5,SELE,MGC=10>               ; 303
006716' 732002  405340
006717' 030403  040000                          MWORD   <JMAP,J=304,D=1>                                ; 304
006720' 001000  000040

                                        ;       Subroutine to insert right 10 bits of R0 into R5 zeroing
                                        ;       the rest of R5.

006721' 060000  310000                          MWORD   <ADDR=600,LDCT,J=31,SOA,OR,B=5,D=2>             ; 600
006722' 432002  400300
006723' 060106  010000                          MWORD   <RPCT,J=601,SOB,B=5,OR,D=5>                     ; 601
006724' 335002  400220
006725' 060200  000000                          MWORD   <CRTN>                                          ; 602
006726' 000000  000240

                                        ;       Subroutine to shift the right justified bit field in R0 to
                                        ;       bits 0-8, all other bits zero.  Then to place in R5 and
                                        ;       in EBUF to prepare for IOP request.  Then request interrupt
                                        ;       and halt.

006727' 070000  000000                          MWORD   <ADDR=700,CONT,SOA,AND,B=5,D=2>                 ; 700
006730' 442002  400340
006731' 070100  320000                          MWORD   <LDCT,J=32,D=1>                                 ; 701
006732' 001000  000300
006733' 070207  020000                          MWORD   <RPCT,J=702,SOA,OR,D=7>                         ; 702
006734' 437000  000220
006735' 070300  002004                          MWORD   <CONT,SAB,B=5,OR,D=3,OENA,SELE,MGC=4>           ; 703
006736' 133002  405340
006737' 070400  311234                          MWORD   <LDCT,J=31,SDO,OR,D=2,B=6,SKCN,MGC=1234>; 704
006740' 732003  240300
006741' 070507  050000                          MWORD   <RPCT,J=705,SOB,B=6,D=5>                        ; 705
006742' 305003  000220
006743' 070600  000001                          MWORD   <CONT,SELE,MGC=1>                               ; 706 (Ex/Dep int)
006744' 000000  005340
006745' 070707  110000                          MWORD   <CJP,J=711,D=1,CENA,CCER>                       ; 707
006746' 001400  100060
006747' 071007  070000                          MWORD   <JMAP,J=707,D=1>                                ; 710
006750' 001000  000040
006751' 071100  002004                          MWORD   <CONT,SOA,OR,A=6,D=1,SELE,MGC=4,OENA>           ; 711
006752' 431060  005340
006753' 071200  000020                          MWORD   <CONT,D=1,SELE,MGC=20>                          ; 712
006754' 001000  005340
006755' 071307  130000                          MWORD   <JMAP,J=713>                                    ; 713
006756' 000000  000040

                                        ;       Subroutine to read R5
```

```
  6377
  6378  006757' 110011  012004             MWORD   <ADDR=1100,JMAP,J=1101,SOA,A=6,OR,D=1,OENA,SELE,MGC=4>
  6379  006760' 431060  005040
  6380  006761' 110111  010000             MWORD   <JMAP,J=1101>                              ; 1101
  6381  006762' 000000  000040
  6382  006763' 777777  777777             -1
```

```

                                        ;#*********************************************************************
                                        ;* TEST 72 - IOP Function 7 - Examine/Increment
                                        ;
                                        ; Description:  This test verifies that microcode can request an
                                        ;               interrupt with an IOP word specifying function 7
                                        ;               and this will cause the  location  specified  by
                                        ;               the  IOP  word  to be incremented in KL10 memory
                                        ;               and examined by the port.
                                        ;
                                        ; Procedure:    KL> Port Clear
                                        ;               KL> Load test ucode
                                        ;               KL> Set up PI system so that interrupts may occur
                                        ;               KL> Set CSR Register PIA bits 33-35 to 0
                                        ;               KL> Write CSR to set MPROC Run
                                        ;               KL> Stop the port, then read the EBUF and verify
                                        ;                   that the data pattern specified is now in
                                        ;                   the EBUF.
                                        ;
                                        ;               Data pattern to use:  125252,,777777
                                        ;
                                        ;               Do for IOP words specifying:
                                        ;
                                        ;                AdrSpa     Func  Qual  Device  Addr   IOP Word
                                        ;                Physical    7     0      0     BUFF  440000,,BUFF
                                        ;                  EPT       7     0      0      100  040000,,000100
                                        ;                Exec Vir    7     0      0      500  140000,,000500
                                        ;                Physical    7     0      0      600  440000,,000600
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

 006764' 254 00 0 00 006771'            TSTE72: JRST    TG72                            ; go start test
 006765' 400400  000072                         EBUS!NDMP!ZEBUS!72                      ; test mask
 006766' 007122' 012542'                        T72M,,[ASCIZ ^IOP Function 7 - Examine/Increment^]
 006767' 012276' 012300'                        [EXP E14,MLAST!E7],,[EXP E4,MLAST!E1]
 006770' 777777  777777                         -1                                      ; failure test table

                                        ; Start test

 006771' 201 00 0 00 000000'            TG72:   MOVEI   Z2                              ; get address of module start
 006772' 260 17 0 00 006531*                    GO      TRACE                           ; handle trace output
 006773' 201 01 0 00 007122'                    MOVEI   1,T72M                          ; set up microcode address
 006774' 260 17 0 00 006533*                    GO      TLOAD                           ; load/verify it
 006775' 263 17 0 00 000000                     RTN                                     ; failed - exit test
```

```

                                          ; Initialization

006776' 201 01 0 00 000001                      MOVEI   1,1                                ; set to 'read'
006777' 201 02 0 00 000010                      MOVEI   2,10                               ; set to 10 words
007000' 201 14 0 00 000010                      MOVEI   PAT,10                             ; set up data pattern
007001' 260 17 0 00 006540*                     GO      BUFGEN                             ; build CCW list (locks BUFF in core)

007002' 400 15 0 00 000000              TL72:   SETZ    ERFLG,                             ; clear error flag
007003' 201 06 0 00 007017'                     MOVEI   6,TS72                             ; get test table address
007004' 402 00 0 00 006543*                     SETZM   TSTSUB                             ; initialize subtest number

                                          ; Loop on test execute table entries

007005' 260 17 0 00 006544*             TA72:   GO      IPACLR                             ; clear port
007006' 260 17 0 00 006545*             TB72:   GO      IEXEC                              ; execute table entry
007007' 254 00 0 00 007016'                       JRST    TX72                             ; end of test execute table
007010' 254 00 0 00 007006'                       JRST    TB72                             ; keep looping after call
007011' 474 15 0 00 000000                        SETO    ERFLG,                           ; error occurred

                                          ; Handle error printouts and scope looping

007012' 027 00 0 00 007120'                     SCOPER  MA72                               ; print error message
007013' 254 00 0 00 007002'                     JRST    TL72                               ; loop on error
007014' 254 00 0 00 007016'                     JRST    TX72                               ; altmode exit
007015' 322 15 0 00 007005'                     JUMPE   ERFLG,TA72                         ; do next test execute table entry

                                          ; End of test

007016' 263 17 0 00 000000              TX72:   RTN                                        ; return

                                          ; Interrupt Test Execute Table, as:  (CMD,parameters)

007017' 040002  007114'                 TS72:   ITABLE  (ISETEX,2,TS72Z)                   ; set up default examine data
007020' 100002  007114'                         ITABLE  (ISETDE,2,TS72Z)                   ; set up default deposit data

007021' 300000  007062'                         ITABLE  (ICALL,TS72EX)                     ; set up test data
007022' 240002  007116'                         ITABLE  (ISETEI,2,TS72AA)                  ; set up examine data
007023' 500000  000100                          ITABLE  (ISTART,100)                       ; start at 100
007024' 400000  000010                          ITABLE  (IWAIT,10)                         ; wait for 10 ms.
007025' 540000  000000                          ITABLE  (IEXIT)                            ; exit test if any error yet

007026' 340000  007036'                         ITABLE  (ICALLC,T72ADI)                    ; initialize start address
007027' 340000  007040'                 TS72A:  ITABLE  (ICALLC,T72ADD)                    ; increment test address
007030' 240004  007116'                         ITABLE  (ISETEI,4,TS72AA)                  ; set up increment/decrement data
007031' 500000  777777                          ITABLE  (ISTART,777777)                    ; start at default address
007032' 400000  000010                          ITABLE  (IWAIT,10)                         ; wait for 10 ms.
007033' 540000  000000                          ITABLE  (IEXIT)                            ; exit test if any error yet
007034' 440000  007027'                         ITABLE  (IJRST,TS72A)                      ; loop till done
007035' 000000  000000                          ITABLE  (ILAST)                            ; end of table

```

```

                                ; Start address handling routines

007036' 476 00 0 00 013673'     T72ADI: SETOM   T72ADR#                 ; initialize start address
007037' 263 17 0 00 000000              RTN                             ; return

007040' 350 01 0 00 013673'     T72ADD: AOS     1,T72ADR                ; increment address
                                        MOVE    [070000,,000100         ; get IOP function word
                                                 170000,,000500
007041' 200 00 0 01 012551'                      470000,,000600](1)
007042' 202 00 0 00 006632*             MOVEM   IIOPF                   ; save it
007043' 200 01 0 00 012554'             MOVE    1,[125252,,777777]      ; initial data
007044' 202 01 0 00 000100              MOVEM   1,100                   ; save it in locations
007045' 202 01 0 00 000500              MOVEM   1,500                   ;   ...
007046' 202 01 0 00 000600              MOVEM   1,600                   ;   ...
007047' 202 01 0 00 007117'             MOVEM   1,TS72AA+1              ; save correct data
007050' 550 00 0 00 007042*             HRRZ    IIOPF                   ; get address to increment
007051' 202 00 0 00 007116'             MOVEM   TS72AA                  ; save it
007052' 200 01 0 00 013673'             MOVE    1,T72ADR                ; get address offset
007053' 200 01 0 01 007110'             MOVE    1,T72ADT(1)             ; use dispatch table
007054' 202 01 0 00 006620*             MOVEM   1,IADDR                 ; set up start address
007055' 201 00 0 00 000010              MOVEI   10                      ; get start CSR data and save in SDATA
007056' 202 00 0 00 006622*             MOVEM   SDATA                   ;   (this ensures PI bits not set)
007057' 302 01 0 00 000777              CAIE    1,777                   ; done yet?
007060' 350 00 0 17 000000              AOS     (P)                     ; no - set up ok return
007061' 263 17 0 00 000000              RTN                             ; return

                                ; Initial examine routine

007062' 200 00 0 00 012555'     TS72EX: MOVE    [470000,,BUFF]          ; get IOP function word
007063' 202 00 0 00 007050*             MOVEM   IIOPF                   ; save it
007064' 200 00 0 00 012554'             MOVE    [125252,,777777]        ; initial data
007065' 202 00 0 00 006631*             MOVEM   BUFF                    ; save it
007066' 202 00 0 00 007117'             MOVEM   TS72AA+1                ; save correct
007067' 550 02 0 00 007063*             HRRZ    2,IIOPF                 ; get address to be examined
007070' 202 02 0 00 007116'             MOVEM   2,TS72AA                ; save it
007071' 336 00 0 00 006646*             SKIPN   UDEBUG                  ; debug mode?
007072' 257 02 0 02 000000              MAP     2,(2)                   ; no - map to physical
007073' 621 02 0 00 777000              TLZ     2,777000                ; mask off unused bits
007074' 201 01 0 00 000300              MOVEI   1,300                   ; get start address
007075' 242 01 0 00 000001              LSH     1,1                     ; convert to address for RAR
007076' 260 17 0 00 006641*             GO      LDRAR                   ; load RAR
007077' 201 01 0 00 000010              MOVEI   1,MPRUN                 ; get start data
007100' 260 17 0 00 006651*             GO      LDCSR                   ; start the port
007101' 200 01 0 00 012303'             MOVE    1,[DATAO 2]             ; build a DATAO
007102' 434 01 0 00 000016              IOR     1,MBCN                  ; include the device code
007103' 336 00 0 00 007071*             SKIPN   UDEBUG                  ; debug mode?
007104' 256 00 0 00 000001              XCT     1                       ; no - do the DATAO
007105' 400 01 0 00 000000              SETZ    1,                      ; clear CSR data
007106' 260 17 0 00 007100*             GO      LDCSR                   ; stop the port
007107' 263 17 0 00 000000              RTN                             ; return
```

```

                                            ; Start address table

007110' 000000  000000                      T72ADT: 0
007111' 000000  000003                              3
007112' 000000  000006                              6
007113' 000000  000777                              777

007114' 000000  000000                      TS72Z:  0
007115' 000000  000000                              0

007116' 000000  000000                      TS72AA: 0
007117' 000000  000000                              0

                                            ; Error messages

007120' 140000  012470'                     MA72:   MSG!TXNOT![ASCIZ /IOP error detected/]
007121' 000000000000#                               LAST!CALL!TXALL!IIPNT           ; print test data

                                            ; Microcode:

                                            ;       IOP = 070000,,000100    EPT / Exam-Incr / Qual=0 / Addr=100

007122' 000000  010000                      T72M:   MWORD   <ADDR=0,JMAP,J=1>                       ; 0
007123' 000000  000040
007124' 000106  000100                              MWORD   <CJS,J=600,SD0,OR,D=2,SKCN,MGC=100>     ; 1
007125' 732000  240020
007126' 000207  010070                              MWORD   <JMAP,J=701,SD0,OR,D=2,SKCN,MGC=70>     ; 2
007127' 732000  240040

                                            ;       IOP = 170000,,000500    Exec Vir / Exam-Incr / Qual=0 / Addr=500

007130' 000300  040000                              MWORD   <JMAP,J=4>                              ; 3
007131' 000000  000040
007132' 000406  000500                              MWORD   <CJS,J=600,SD0,OR,D=2,SKCN,MGC=500>     ; 4
007133' 732000  240020
007134' 000507  010170                              MWORD   <JMAP,J=701,SD0,OR,D=2,SKCN,MGC=170>    ; 5
007135' 732000  240040

                                            ;       IOP = 470000,,000600    Physical / Exam-Incr / Qual=0 / Addr=600

007136' 000600  070000                              MWORD   <JMAP,J=7>                              ; 6
007137' 000000  000040
007140' 000706  000600                              MWORD   <CJS,J=600,SD0,OR,D=2,SKCN,MGC=600>     ; 7
007141' 732000  240020
007142' 001007  010470                              MWORD   <JMAP,J=701,SD0,OR,D=2,SKCN,MGC=470>    ; 10
007143' 732000  240040

                                            ;       IOP = 470000,,BUFF      Physical / Examine-Inc / Qual=0 / Addr=BUFF

007144' 010001  010000                              MWORD   <ADDR=100,JMAP,J=101>                   ; 100
007145' 000000  000040
007146' 010107  010470                              MWORD   <JMAP,J=701,SD0,OR,D=2,SKCN,MGC=470>    ; 101
007147' 732000  240040

```

```
                                  ;       Routine to read the buffer address and save in R5

007150' 030003  010000                    MWORD   <ADDR=300,JMAP,J=301>                         ; 300
007151' 000000  000040
007152' 030103  030000                    MWORD   <CJP,J=303,CENA,CCER>                         ; 301
007153' 000400  100060
007154' 030203  010000                    MWORD   <JMAP,J=301>                                  ; 302
007155' 000000  000040
007156' 030300  000010                    MWORD   <CONT,SDO,OR,D=2,B=5,SELE,MGC=10>             ; 303
007157' 732002  405340
007160' 030403  040000                    MWORD   <JMAP,J=304,D=1>                              ; 304
007161' 001000  000040

                                  ;       Subroutine to insert right 10 bits of R0 into R5 zeroing
                                  ;       the rest of R5.

007162' 060000  310000                    MWORD   <ADDR=600,LDCT,J=31,SOA,OR,B=5,D=2>           ; 600
007163' 432002  400300
007164' 060106  010000                    MWORD   <RPCT,J=601,SOB,B=5,OR,D=5>                   ; 601
007165' 335002  400220
007166' 060200  000000                    MWORD   <CRTN>                                        ; 602
007167' 000000  000240

                                  ;       Subroutine to shift the right justified bit field in R0 to
                                  ;       bits 0-8, all other bits zero.  Then request interrupt and
                                  ;       wait for CC EbusRqst.  Then put EBUS data into R5 and halt.

007170' 070000  000000                    MWORD   <ADDR=700,CONT,SOA,AND,B=5,D=2>               ; 700
007171' 442002  400340
007172' 070100  320000                    MWORD   <LDCT,J=32,D=1>                               ; 701
007173' 001000  000300
007174' 070207  020000                    MWORD   <RPCT,J=702,SOA,OR,D=7>                       ; 702
007175' 437000  000220
007176' 070300  002004                    MWORD   <CONT,SAB,OR,B=5,D=3,OENA,SELE,MGC=4>         ; 703
007177' 133002  405340
007200' 070400  000001                    MWORD   <CONT,SELE,MGC=1>                             ; 704 (Ex/Dep int)
007201' 000000  005340
007202' 070507  070000                    MWORD   <CJP,J=707,D=1,CENA,CCER>                     ; 705
007203' 001400  100060
007204' 070607  050000                    MWORD   <JMAP,J=705,D=1>                              ; 706
007205' 001000  000040
007206' 070700  000010                    MWORD   <CONT,SDO,OR,D=2,B=5,SELE,MGC=10>             ; 707
007207' 732002  405340
007210' 071007  100000                    MWORD   <JMAP,J=710>                                  ; 710
007211' 000000  000040

                                  ;       Subroutine to read R5

007212' 110011  012004                    MWORD   <ADDR=1100,JMAP,J=1101,SOA,A=5,OR,D=1,OENA,SELE,MGC=4>
007213' 431050  005040
007214' 110111  010000                    MWORD   <JMAP,J=1101>                                 ; 1101
007215' 000000  000040
007216' 777777  777777                    -1
```

```

                                ;#*********************************************************************
                                ;* TEST 73 - CSR 25-26 & 28 KL-side
                                ;
                                ; Description:  Verify  that  CSR  Bits 25-26 and 28 can be read
                                ;               and written properly from the KL side.
                                ;
                                ; Procedure:    KL> Load test ucode
                                ;
                                ;               KL> EBUS Reset
                                ;               KL> Read CSR and verify that bits 25-26,28 are 0's
                                ;
                                ;               KL> Write 0's to bits 25-26,28
                                ;               KL> Read CSR and verify that bits 25-26,28 are 0's
                                ;
                                ;               KL> Write 1's to bits 25-26,28
                                ;               KL> Read CSR and verify that bits 25-26,28 are 0's
                                ;
                                ;               KL> Start up port at location 0
                                ;               UC> Write 1's to bits 25-26,28
                                ;               KL> Read CSR and verify that bits 25-26,28 are 1's
                                ;
                                ;               KL> Write 1's to bits 25-26,28
                                ;               KL> Read CSR and verify that bits 25-26,28 are 0's
                                ;
                                ;               KL> Start up port at location 0
                                ;               UC> Write 1's to bits 25-26,28
                                ;               KL> Do an EBUS Reset
                                ;               KL> Read CSR and verify that bits 25-26,28 are 0's
                                ;
                                ; Failure:      ---
                                ;#*********************************************************************

                                ; Test data

007217' 254 00 0 00 007227'     TSTE73: JRST    TG73                    ; go start test
007220' 400400  000073                  EBUS!NDMP!ZEBUS!73              ; test mask
007221' 007361' 012556'                 T73M,,[ASCIZ ^CSR 25-26 & 28 KL-side^]
007222' 010761' 012563'                 [EXP MLAST!E11],,[EXP E9,E1,MLAST!E6]
007223' 000000  007402'                 TSTE74                          ; failure test table
007224' 000000  007613'                 TSTE75                          ;   ...
007225' 000000  007744'                 TSTE76
007226' 777777  777777                  -1

                                ; Start test

007227' 201 00 0 00 000000'     TG73:   MOVEI   Z2                      ; get address of module start
007230' 260 17 0 00 006772*             GO      TRACE                   ; handle trace output
007231' 201 01 0 00 007361'             MOVEI   1,T73M                  ; set up microcode address
007232' 260 17 0 00 006774*             GO      TLOAD                   ; load/verify it
007233' 263 17 0 00 000000              RTN                             ; failed - exit test
```

```

                                    ; KL> EBUS Reset
                                    ; KL> Read CSR and verify that bits 25-26,28 are 0's

007234' 400 15 0 00 000000          TA73:   SETZ    ERFLG,                          ; clear error flag
007235' 260 17 0 00 004716*                 GO      ERESET                          ; do an EBUS Reset
007236' 201 00 0 00 000001                  MOVEI   1                               ; initialize subtest number
007237' 202 00 0 00 007004*                 MOVEM   TSTSUB                          ;   to 1
007240' 260 17 0 00 005556*                 GO      RDCSR                           ; read CSR register
007241' 255 00 0 00 000000                  JFCL                                    ; error - ignore
007242' 603 01 0 00 003200                  TLNE    1,3200                          ; bits all zero?
007243' 474 15 0 00 000000                  SETO    ERFLG,                          ; no - set error flag
007244' 027 00 0 00 007346'                 SCOPER  MB73                            ; print error message
007245' 254 00 0 00 007234'                 JRST    TA73                            ; loop on error
007246' 254 00 0 00 007345'                 JRST    TX73                            ; altmode exit
007247' 326 15 0 00 007345'                 JUMPN   ERFLG,TX73                      ; error? yes - exit

                                    ; KL> Write 0's to bits 25-26,28
                                    ; KL> Read CSR and verify that bits 25-26,28 are 0's

007250' 350 00 0 00 007237*                 AOS     TSTSUB                          ; indicate next subtest
007251' 400 01 0 00 000000                  SETZ    1,                              ; clear data
007252' 260 17 0 00 007106*                 GO      LDCSR                           ; write CSR register
007253' 260 17 0 00 007240*                 GO      RDCSR                           ; read CSR register
007254' 255 00 0 00 000000                  JFCL                                    ; error - ignore
007255' 603 01 0 00 003200                  TLNE    1,3200                          ; bits all zero?
007256' 474 15 0 00 000000                  SETO    ERFLG,                          ; no - set error flag
007257' 027 00 0 00 007346'                 SCOPER  MB73                            ; print error message
007260' 254 00 0 00 007234'                 JRST    TA73                            ; loop on error
007261' 254 00 0 00 007345'                 JRST    TX73                            ; altmode exit
007262' 326 15 0 00 007345'                 JUMPN   ERFLG,TX73                      ; error? yes - exit

                                    ; KL> Write 1's to bits 25-26,28
                                    ; KL> Read CSR and verify that bits 25-26,28 are 0's

007263' 350 00 0 00 007250*                 AOS     TSTSUB                          ; indicate next subtest
007264' 201 01 0 00 003200                  MOVEI   1,3200                          ; get 1's for bits
007265' 260 17 0 00 007252*                 GO      LDCSR                           ; write CSR register
007266' 260 17 0 00 007253*                 GO      RDCSR                           ; read CSR register
007267' 255 00 0 00 000000                  JFCL                                    ; error - ignore
007270' 603 01 0 00 003200                  TLNE    1,3200                          ; bits all zero?
007271' 474 15 0 00 000000                  SETO    ERFLG,                          ; no - set error flag
007272' 027 00 0 00 007346'                 SCOPER  MB73                            ; print error message
007273' 254 00 0 00 007234'                 JRST    TA73                            ; loop on error
007274' 254 00 0 00 007345'                 JRST    TX73                            ; altmode exit
007275' 326 15 0 00 007345'                 JUMPN   ERFLG,TX73                      ; error? yes - exit
```

```

                                        ; KL> Start up port at location 0
                                        ; UC> Write 1's to bits 25-26,28
                                        ; KL> Read CSR and verify that bits 25-26,28 are 1's

007276' 350 00 0 00 007263*                     AOS     TSTSUB                  ; indicate next subtest
007277' 400 01 0 00 000000                      SETZ    1,                      ; set up address 0
007300' 260 17 0 00 007076*                     GO      LDRAR                   ; write to RAR
007301' 201 01 0 00 000010                      MOVEI   1,10                    ; get 1's for bits
007302' 260 17 0 00 007265*                     GO      LDCSR                   ; write CSR register
007303' 260 17 0 00 007266*                     GO      RDCSR                   ; read CSR register
007304' 255 00 0 00 000000                      JFCL                            ; error - ignore
007305' 135 00 0 00 012566'                     LDB     [POINT  4,1,28]         ; get bits
007306' 302 00 0 00 000015                      CAIE    15                      ; all set?
007307' 474 15 0 00 000000                      SETO    ERFLG,                  ; no - set error flag
007310' 027 00 0 00 007346'                     SCOPER  MB73                    ; print error message
007311' 254 00 0 00 007234'                     JRST    TA73                    ; loop on error
007312' 254 00 0 00 007345'                     JRST    TX73                    ; altmode exit
007313' 326 15 0 00 007345'                     JUMPN   ERFLG,TX73              ; error? yes - exit

                                        ; KL> Write 1's to bits 25-26,28
                                        ; KL> Read CSR and verify that bits 25-26,28 are 0's

007314' 350 00 0 00 007276*                     AOS     TSTSUB                  ; indicate next subtest
007315' 201 01 0 00 003200                      MOVEI   1,3200                  ; get 1's for bits
007316' 260 17 0 00 007302*                     GO      LDCSR                   ; write CSR register
007317' 260 17 0 00 007303*                     GO      RDCSR                   ; read CSR register
007320' 255 00 0 00 000000                      JFCL                            ; error - ignore
007321' 603 01 0 00 003200                      TLNE    1,3200                  ; bits all zero?
007322' 474 15 0 00 000000                      SETO    ERFLG,                  ; no - set error flag
007323' 027 00 0 00 007346'                     SCOPER  MB73                    ; print error message
007324' 254 00 0 00 007234'                     JRST    TA73                    ; loop on error
007325' 254 00 0 00 007345'                     JRST    TX73                    ; altmode exit
007326' 326 15 0 00 007345'                     JUMPN   ERFLG,TX73              ; error? yes - exit

                                        ; KL> Start up port at location 0
                                        ; UC> Write 1's to bits 25-26,28
                                        ; KL> Do an EBUS Reset
                                        ; KL> Read CSR and verify that bits 25-26,28 are 0's

007327' 350 00 0 00 007314*                     AOS     TSTSUB                  ; indicate next subtest
007330' 400 01 0 00 000000                      SETZ    1,                      ; set up address 0
007331' 260 17 0 00 007300*                     GO      LDRAR                   ; write to RAR
007332' 201 01 0 00 000010                      MOVEI   1,10                    ; get 1's for bits
007333' 260 17 0 00 007316*                     GO      LDCSR                   ; write CSR register
007334' 260 17 0 00 007235*                     GO      ERESET                  ; do an EBUS reset
007335' 260 17 0 00 007317*                     GO      RDCSR                   ; read CSR register
007336' 255 00 0 00 000000                      JFCL                            ; error - ignore
007337' 603 01 0 00 003200                      TLNE    1,3200                  ; bits all zero?
007340' 474 15 0 00 000000                      SETO    ERFLG,                  ; no - set error flag
007341' 027 00 0 00 007346'                     SCOPER  MB73                    ; print error message
007342' 254 00 0 00 007234'                     JRST    TA73                    ; loop on error
007343' 254 00 0 00 007345'                     JRST    TX73                    ; altmode exit
007344' 326 15 0 00 007345'                     JUMPN   ERFLG,TX73              ; error? yes - exit
```

```
                                   ; End of test

007345' 263 17 0 00 000000         TX73:   RTN                          ; return

                                   ; Error messages

007346' 270000  007347'            MB73:   LAST!CALL!TXALL!MB73PN       ; print data

007347' 200 02 0 00 007327*        MB73PN: MOVE     2,TSTSUB            ; get subtest number
007350' 256 00 0 02 007352'                XCT      MB73PP-1(2)         ; print text
007351' 260 17 0 00 010231'                GO       MPNT2               ; print CSR data
007352' 263 17 0 00 000000                 RTN

                                   MB73PP: TMSG    <
                                   Did EBUS Reset
007353' 037 00 0 00 012567'        CSR Bits 25-26 & 28 not zero>
                                           TMSG    <
                                   Did EBUS Reset, then wrote 0's to CSR
007354' 037 00 0 00 012601'        CSR Bits 25-26 & 28 not zero>
                                           TMSG    <
                                   Did EBUS Reset, then wrote 1's to CSR
007355' 037 00 0 00 012617'        CSR Bits 25-26 & 28 not zero>
                                           TMSG    <
                                   Port wrote 1's to CSR bits 25-26 & 28, then KL read CSR
007356' 037 00 0 00 012635'        CSR Bits 25-26 & 28 not set>
                                           TMSG    <
                                   Port wrote 1's to CSR bits 25-26 & 28, then KL wrote 1's to CSR
007357' 037 00 0 00 012657'        CSR Bits 25-26 & 28 not cleared>
                                           TMSG    <
                                   Port wrote 1's to CSR bits 25-26 & 28, then KL did an EBUS Reset
007360' 037 00 0 00 012703'        CSR Bits 25-26 & 28 not cleared>

                                   ; Microcode:

007361' 000000  010015             T73M:   MWORD   <ADDR=0,JMAP,J=1,SDO,OR,D=2,SKCN,MGC=15>      ; 0
007362' 732000  240040
007363' 000100  220000                     MWORD   <LDCT,J=22,D=1>                               ; 1
007364' 001000  000300
007365' 000200  020000                     MWORD   <RPCT,J=2,SOA,OR,D=5>                         ; 2
007366' 435000  000220
007367' 000300  002004                     MWORD   <CONT,SOA,OR,D=1,OENA,SELE,MGC=4>             ; 3
007370' 431000  005340
007371' 000400  060040                     MWORD   <CJP,J=6,CENA,CCGC,SELE,MGC=40>               ; 4
007372' 000400  015060
007373' 000500  040000                     MWORD   <JMAP,J=4>                                    ; 5
007374' 000000  000040
007375' 000600  000200                     MWORD   <CONT,SOA,A=1,OR,SELE,MGC=200>                ; 6
007376' 430010  005340
007377' 000700  070000                     MWORD   <JMAP,J=7>                                    ; 7
007400' 000000  000040
007401' 777777  777777                     -1
```

```
                                        ;#*********************************************************************
                                        ;* TEST 74 - CSR 25-26 & 28 Port-side
                                        ;
                                        ; Description:  Verify  that  CSR  Bits 25-26 and 28 can be read
                                        ;               and written properly from the port side.
                                        ;
                                        ; Procedure:    KL> Load test ucode
                                        ;               KL> EBUS Reset
                                        ;               KL> Start port at location 0
                                        ;
                                        ;               UC> Read CSR and verify that bits 25-26,28 are 0's
                                        ;                   If error - set completion code in EBUF of 1
                                        ;
                                        ;               UC> Write 0's to bits 25-26,28
                                        ;               UC> Read CSR and verify that bits 25-26,28 are 0's
                                        ;                   If error - set completion code in EBUF of 2
                                        ;
                                        ;               UC> Write 1's to bits 25-26,28
                                        ;               UC> Read CSR and verify that bits 25-26,28 are 1's
                                        ;                   If error - set completion code in EBUF of 3
                                        ;
                                        ;               UC> Write 0's to bits 25-26,28
                                        ;               UC> Read CSR and verify that bits 25-26,28 are 1's
                                        ;                   If error - set completion code in EBUF of 4
                                        ;
                                        ;               If no errors - set completion code in EBUF of 0
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

007402' 254 00 0 00 007411'             TSTE74: JRST    TG74                            ; go start test
007403' 400400  000074                          EBUS!NDMP!ZEBUS!74                      ; test mask
007404' 007462' 012727'                         T74M,,[ASCIZ ^CSR 25-26 & 28 Port-side^]
007405' 010761' 012563'                         [EXP MLAST!E11],,[EXP E9,E1,MLAST!E6]
007406' 000000  007613'                         TSTE75                                  ; failure test table
007407' 000000  007744'                         TSTE76                                  ;  ...
007410' 777777  777777                          -1

                                        ; Start test

007411' 201 00 0 00 000000'             TG74:   MOVEI   Z2                              ; get address of module start
007412' 260 17 0 00 007230*                     GO      TRACE                           ; handle trace output
007413' 201 01 0 00 007462'                     MOVEI   1,T74M                          ; set up microcode address
007414' 260 17 0 00 007232*                     GO      TLOAD                           ; load/verify it
007415' 263 17 0 00 000000                      RTN                                     ; failed - exit test
```

```
                                      ; KL> EBUS Reset
                                      ; KL> Start port at location 0

 007416' 400 15 0 00 000000     TA74:   SETZ    ERFLG.                  ; clear error flag
 007417' 260 17 0 00 007334*            GO      ERESET                  ; do an EBUS Reset
 007420' 201 00 0 00 000001             MOVEI   1                       ; set subtest number
 007421' 202 00 0 00 007347*            MOVEM   TSTSUB                  ;   to 1
 007422' 201 00 0 00 000014             MOVEI   MPRUN!4                 ; get startup bits
 007423' 202 00 0 00 007056*            MOVEM   SDATA                   ; save in startup data word
 007424' 402 00 0 00 005531*            SETZM   SNEXT                   ; set start address to 0
 007425' 260 17 0 00 010277'            GO      MSTART                  ; start up port
 007426' 254 00 0 00 007416'            JRST    TA74                    ; loop on error
 007427' 254 00 0 00 007445'            JRST    TX74                    ; altmode exit
 007430' 326 15 0 00 007445'            JUMPN   ERFLG,TX74              ; error? yes - exit

                                ; Port does the remainder of the testing and returns completion code
                                ; in the EBUF.

 007431' 350 00 0 00 007421*            AOS     TSTSUB                  ; set up subtest number
 007432' 201 01 0 00 200000             MOVEI   1,TSTEBF                ; set up to read EBUF
 007433' 260 17 0 00 007333*            GO      LDCSR                   ; stop the port
 007434' 260 17 0 00 003274*            GO      RDEBUF                  ; read completion code
 007435' 405 01 0 00 000007             ANDI    1,7                     ; mask out irrelevant bits
 007436' 202 01 0 00 013665'            MOVEM   1,SAVEBU#               ; save completion code
 007437' 332 00 0 00 000001             SKIPE   1                       ; error code set?
 007440' 474 15 0 00 000000             SETO    ERFLG.                  ; yes - set error flag
 007441' 027 00 0 00 007446'            SCOPER  MB74                    ; print error message
 007442' 254 00 0 00 007416'            JRST    TA74                    ; loop on error
 007443' 254 00 0 00 007445'            JRST    TX74                    ; altmode exit
 007444' 326 15 0 00 007445'            JUMPN   ERFLG,TX74              ; error? yes - exit

                                ; End of test

 007445' 263 17 0 00 000000     TX74:   RTN                             ; return

                                ; Error messages

 007446' 270000  007447'        MB74:   LAST!CALL!TXALL!MB74PN          ; print data

 007447' 200 02 0 00 013665'    MB74PN: MOVE    2,SAVEBU                ; get subtest number
 007450' 256 00 0 02 007452'            XCT     MB74PP-1(2)             ; print text
 007451' 260 17 0 00 010231'            GO      MPNT2                   ; print CSR data
 007452' 263 17 0 00 000000             RTN
```

```

                                      MB74PP: TMSG    <
                                      Did EBUS Reset, Port read CSR
007453' 037 00 0 00 012734'           CSR Bits 25-26 & 28 not zero>
                                              TMSG    <
                                      Did EBUS Reset, Port wrote 0's to CSR
007454' 037 00 0 00 012751'           CSR Bits 25-26 & 28 not zero>
                                              TMSG    <
                                      Port wrote 1's to CSR
007455' 037 00 0 00 012767'           CSR Bits 25-26 & 28 not 1's>
                                              TMSG    <
                                      Port wrote 1's to CSR
                                      Port wrote 0's to CSR
007456' 037 00 0 00 013002'           CSR Bits 25-26 & 28 not 1's>
007457' 037 00 0 00 013022'                   TMSGC   <Failure code incorrect - 5>
007460' 037 00 0 00 013030'                   TMSGC   <Failure code incorrect - 6>
007461' 037 00 0 00 013036'                   TMSGC   <Failure code incorrect - 7>

                                      ; Microcode:    Build pattern in R2

007462' 000000  010015                T74M:   MWORD   <ADDR=0,JMAP,J=1,SDO,OR,B=2,D=2,SKCN,MGC=15>    ; 0
007463' 732001  240040
007464' 000100  310000                        MWORD   <LDCT,J=31>                                     ; 1
007465' 000000  000300
007466' 000200  020000                        MWORD   <RPCT,J=2,SOB,OR,B=2,D=5>                       ; 2
007467' 335001  000220
007470' 000300  060000                        MWORD   <LDCT,J=6>                                      ; 3
007471' 000000  000300
007472' 000400  040000                        MWORD   <RPCT,J=4,SOB,OR,B=2,D=7>                       ; 4
007473' 337001  000220

                                      ;       Read CSR and verify that bits 25-26,28 are 0's

007474' 000500  070040                        MWORD   <CJP,J=7,CENA,CCGC,SELE,MGC=40>                 ; 5
007475' 000400  015060
007476' 000600  050000                        MWORD   <JMAP,J=5>                                      ; 6
007477' 000000  000040
007500' 000700  000100                        MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>              ; 7
007501' 732000  405340
007502' 001000  000000                        MWORD   <CONT,SAB,A=2,B=1,AND,D=2>                      ; 10
007503' 142020  400340
007504' 001100  130000                        MWORD   <CJP,J=13,SAB,AND,A=1,B=2,D=1,CENA,CCFZ>        ; 11
007505' 141411  020060
007506' 001201  000001                        MWORD   <JMAP,J=100,SDO,OR,D=2,SKCN,MGC=1>              ; 12
007507' 732000  240040

                                      ;       UC> Write 0's to bits 25-26,28
                                      ;       UC> Read CSR and verify that bits 25-26,28 are 0's

007510' 001300  002004                        MWORD   <CONT,SOA,AND,D=1,OENA,SELE,MGC=4>              ; 13
007511' 441000  005340
007512' 001400  160040                        MWORD   <CJP,J=16,CENA,CCGC,SELE,MGC=40>                ; 14
007513' 000400  015060
007514' 001500  140000                        MWORD   <JMAP,J=14>                                     ; 15
007515' 000000  000040
```

```
 6990  007516' 001600  000200                      MWORD   <CONT,SELE,MGC=200>                              ; 16
 6991  007517' 000000  005340
 6992
 6993  007520' 001700  210040                      MWORD   <CJP,J=21,CENA,CCGC,SELE,MGC=40>                 ; 17
 6994  007521' 000400  015060
 6995  007522' 002000  170000                      MWORD   <JMAP,J=17>                                      ; 20
 6996  007523' 000000  000040
 6997  007524' 002100  000100                      MWORD   <CONT,SD0,B=1,OR,D=2,SELE,MGC=100>               ; 21
 6998  007525' 732000  405340
 6999  007526' 002200  000000                      MWORD   <CONT,SAB,A=2,B=1,AND,D=2>                       ; 22
 7000  007527' 142020  400340
 7001  007530' 002300  250000                      MWORD   <CJP,J=25,SAB,AND,A=1,B=2,D=1,CENA,CCFZ>         ; 23
 7002  007531' 141411  020060
 7003  007532' 002401  000002                      MWORD   <JMAP,J=100,SD0,OR,D=2,SKCN,MGC=2>               ; 24
 7004  007533' 732000  240040
 7005
 7006                                  ;           UC> Write 1's to bits 25-26,28
 7007                                  ;           UC> Read CSR and verify that bits 25-26,28 are 1's
 7008
 7009  007534' 002500  002004                      MWORD   <CONT,SOA,OR,A=2,D=1,OENA,SELE,MGC=4>            ; 25
 7010  007535' 431020  005340
 7011  007536' 002600  300040                      MWORD   <CJP,J=30,CENA,CCGC,SELE,MGC=40>                 ; 26
 7012  007537' 000400  015060
 7013  007540' 002700  260000                      MWORD   <JMAP,J=26>                                      ; 27
 7014  007541' 000000  000040
 7015  007542' 003000  000200                      MWORD   <CONT,SELE,MGC=200>                              ; 30
 7016  007543' 000000  005340
 7017
 7018  007544' 003100  330040                      MWORD   <CJP,J=33,CENA,CCGC,SELE,MGC=40>                 ; 31
 7019  007545' 000400  015060
 7020  007546' 003200  310000                      MWORD   <JMAP,J=31>                                      ; 32
 7021  007547' 000000  000040
 7022  007550' 003300  000100                      MWORD   <CONT,SD0,B=1,OR,D=2,SELE,MGC=100>               ; 33
 7023  007551' 732000  405340
 7024  007552' 003400  000000                      MWORD   <CONT,SAB,A=2,B=1,AND,D=2>                       ; 34
 7025  007553' 142020  400340
 7026  007554' 003500  370000                      MWORD   <CJP,J=37,SAB,XOR,A=1,B=2,D=1,CENA,CCFZ>         ; 35
 7027  007555' 161411  020060
 7028  007556' 003601  000003                      MWORD   <JMAP,J=100,SD0,OR,D=2,SKCN,MGC=3>               ; 36
 7029  007557' 732000  240040
 7030
 7031                                  ;           UC> Write 0's to bits 25-26,28
 7032                                  ;           UC> Read CSR and verify that bits 25-26,28 are 1's
 7033
 7034  007560' 003700  002004                      MWORD   <CONT,SOA,AND,A=2,D=1,OENA,SELE,MGC=4>           ; 37
 7035  007561' 441020  005340
 7036  007562' 004000  420040                      MWORD   <CJP,J=42,CENA,CCGC,SELE,MGC=40>                 ; 40
 7037  007563' 000400  015060
 7038  007564' 004100  400040                      MWORD   <JMAP,J=40>                                      ; 41
 7039  007565' 000000  000040
 7040  007566' 004200  000200                      MWORD   <CONT,SELE,MGC=200>                              ; 42
 7041  007567' 000000  005340
 7042
 7043  007570' 004300  450040                      MWORD   <CJP,J=45,CENA,CCGC,SELE,MGC=40>                 ; 43
 7044  007571' 000400  015060
 7045  007572' 004400  430000                    MWORD   <JMAP,J=43>                                   ; 44
 7046  007573' 000000  000040
 7047  007574' 004500  000100                    MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>            ; 45
 7048  007575' 732000  405340
 7049  007576' 004600  000000                    MWORD   <CONT,SAB,A=2,B=1,AND,D=2>                    ; 46
 7050  007577' 142020  400340
 7051  007600' 004700  510000                    MWORD   <CJP,J=51,SAB,XOR,A=1,B=2,D=1,CENA,CCFZ>      ; 47
 7052  007601' 161411  020060
 7053  007602' 005001  000004                    MWORD   <JMAP,J=100,SDO,OR,D=2,SKCN,MGC=4>            ; 50
 7054  007603' 732000  240040
 7055  007604' 005101  000000                    MWORD   <JMAP,J=100,SDO,OR,D=2,SKCN,MGC=0>            ; 51
 7056  007605' 732000  240040
 7057
 7058                                    ;       UC> Write completion code
 7059
 7060  007606' 010000  002004                    MWORD   <ADDR=100,CONT,SOA,OR,D=1,OENA,SELE,MGC=4>    ; 100
 7061  007607' 431000  005340
 7062  007610' 010101  010000                    MWORD   <JMAP,J=101,D=1>                              ; 101
 7063  007611' 001000  000040
 7064  007612' 777777  777777                    -1
```

```
                                      ;#*********************************************************************
                                      ;* TEST 75 - CSR 27 & 30-31 KL-side
                                      ;
                                      ; Description:   Verify  that  CSR  Bits 27 and 30-31 can be read
                                      ;                and written properly from the KL side.
                                      ;
                                      ; Procedure:     KL> EBUS Reset
                                      ;                KL> Read CSR and verify that bits 27,30-31 are 0's
                                      ;
                                      ;                KL> Write 0's to bits 27,30-31
                                      ;                KL> Read CSR and verify that bits 27,30-31 are 0's
                                      ;
                                      ;                KL> Write 1's to bits 27,30-31
                                      ;                KL> Read CSR and verify that bits 27,30-31 are 1's
                                      ;
                                      ;                KL> Write 0's to bits 27,30-31
                                      ;                KL> Read CSR and verify that bits 27,30-31 are 1's
                                      ;
                                      ;                KL> Write 1's to bits 27,30-31
                                      ;                KL> Read CSR and verify that bits 27,30-31 are 1's
                                      ;
                                      ;                KL> Do an EBUS Reset
                                      ;                KL> Read CSR and verify that bits 27,30-31 are 0's
                                      ;
                                      ; Failure:       ---
                                      ;#*********************************************************************

                                      ; Test data

007613' 254 00 0 00 007621'           TSTE75: JRST    TG75                    ; go start test
007614' 400000  000075                        EBUS!ZEBUS!75                   ; test mask
007615' 000000  013044'                       0,,[ASCIZ ^CSR 27 & 30-31 KL-side^]
007616' 010761' 012563'                       [EXP MLAST!E11],,[EXP E9,E1,MLAST!E6]
007617' 000000  007744'                       TSTE76                          ; failure test table
007620' 777777  777777                        -1

                                      ; Start test

007621' 201 00 0 00 000000'           TG75:   MOVEI   Z2                      ; get address of module start
007622' 260 17 0 00 007412*                   GO      TRACE                   ; handle trace output
```

```

                                        ; KL> EBUS Reset
                                        ; KL> Read CSR and verify that bits 27,30-31 are 0's

007623' 400 15 0 00 000000      TA75:   SETZ    ERFLG,                  ; clear error flag
007624' 260 17 0 00 007417*             GO      ERESET                  ; do an EBUS Reset
007625' 201 00 0 00 000001              MOVEI   1                       ; initialize subtest number
007626' 202 00 0 00 007431*             MOVEM   TSTSUB                  ;   to 1
007627' 260 17 0 00 007335*             GO      RDCSR                   ; read CSR register
007630' 255 00 0 00 000000              JFCL                            ; error - ignore
007631' 603 01 0 00 000460              TLNE    1,460                   ; bits all zero?
007632' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
007633' 027 00 0 00 007731'             SCOPER  MB75                    ; print error message
007634' 254 00 0 00 007623'             JRST    TA75                    ; loop on error
007635' 254 00 0 00 007730'             JRST    TX75                    ; altmode exit
007636' 326 15 0 00 007730'             JUMPN   ERFLG,TX75              ; error? yes - exit

                                        ; KL> Write 0's to bits 27,30-31
                                        ; KL> Read CSR and verify that bits 27,30-31 are 0's

007637' 350 00 0 00 007626*             AOS     TSTSUB                  ; indicate next subtest
007640' 400 01 0 00 000000              SETZ    1,                      ; clear data
007641' 260 17 0 00 007433*             GO      LDCSR                   ; write CSR register
007642' 260 17 0 00 007627*             GO      RDCSR                   ; read CSR register
007643' 255 00 0 00 000000              JFCL                            ; error - ignore
007644' 603 01 0 00 000460              TLNE    1,460                   ; bits all zero?
007645' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
007646' 027 00 0 00 007731'             SCOPER  MB75                    ; print error message
007647' 254 00 0 00 007623'             JRST    TA75                    ; loop on error
007650' 254 00 0 00 007730'             JRST    TX75                    ; altmode exit
007651' 326 15 0 00 007730'             JUMPN   ERFLG,TX75              ; error? yes - exit

                                        ; KL> Write 1's to bits 27,30-31
                                        ; KL> Read CSR and verify that bits 27,30-31 are 1's

007652' 350 00 0 00 007637*             AOS     TSTSUB                  ; indicate next subtest
007653' 201 01 0 00 000460              MOVEI   1,460                   ; get 1's for bits
007654' 260 17 0 00 007641*             GO      LDCSR                   ; write CSR register
007655' 260 17 0 00 007642*             GO      RDCSR                   ; read CSR register
007656' 255 00 0 00 000000              JFCL                            ; error - ignore
007657' 135 00 0 00 013051'             LDB     [POINT 5,1,31]          ; get bits
007660' 302 00 0 00 000023              CAIE    23                      ; still set?
007661' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
007662' 027 00 0 00 007731'             SCOPER  MB75                    ; print error message
007663' 254 00 0 00 007623'             JRST    TA75                    ; loop on error
007664' 254 00 0 00 007730'             JRST    TX75                    ; altmode exit
007665' 326 15 0 00 007730'             JUMPN   ERFLG,TX75              ; error? yes - exit
```

```

                                        ; KL> Write 0's to bits 27,30-31
                                        ; KL> Read CSR and verify that bits 27,30-31 are 1's

007666' 350 00 0 00 007652*                     AOS     TSTSUB                  ; indicate next subtest
007667' 400 01 0 00 000000                      SETZ    1,                      ; zero AC1
007670' 260 17 0 00 007654*                     GO      LDCSR                   ; zero CSR register
007671' 260 17 0 00 007655*                     GO      RDCSR                   ; read CSR register
007672' 255 00 0 00 000000                      JFCL                            ; error - ignore
007673' 135 00 0 00 013051'                     LDB     [POINT 5,1,31]          ; get bits
007674' 302 00 0 00 000023                      CAIE    23                      ; still set?
007675' 474 15 0 00 000000                      SETO    ERFLG,                  ; no - set error flag
007676' 027 00 0 00 007731'                     SCOPER  MB75                    ; print error message
007677' 254 00 0 00 007623'                     JRST    TA75                    ; loop on error
007700' 254 00 0 00 007730'                     JRST    TX75                    ; altmode exit
007701' 326 15 0 00 007730'                     JUMPN   ERFLG,TX75              ; error? yes - exit

                                        ; KL> Write 1's to bits 27,30-31
                                        ; KL> Read CSR and verify that bits 27,30-31 are 1's

007702' 350 00 0 00 007666*                     AOS     TSTSUB                  ; indicate next subtest
007703' 201 01 0 00 000460                      MOVEI   1,460                   ; get 1's for bits
007704' 260 17 0 00 007670*                     GO      LDCSR                   ; write CSR register
007705' 260 17 0 00 007671*                     GO      RDCSR                   ; read CSR register
007706' 255 00 0 00 000000                      JFCL                            ; error - ignore
007707' 135 00 0 00 013051'                     LDB     [POINT 5,1,31]          ; get bits
007710' 302 00 0 00 000023                      CAIE    23                      ; still set?
007711' 474 15 0 00 000000                      SETO    ERFLG,                  ; no - set error flag
007712' 027 00 0 00 007731'                     SCOPER  MB75                    ; print error message
007713' 254 00 0 00 007623'                     JRST    TA75                    ; loop on error
007714' 254 00 0 00 007730'                     JRST    TX75                    ; altmode exit
007715' 326 15 0 00 007730'                     JUMPN   ERFLG,TX75              ; error? yes - exit

                                        ; KL> Do an EBUS Reset
                                        ; KL> Read CSR and verify that bits 27,30-31 are 0's

007716' 350 00 0 00 007702*                     AOS     TSTSUB                  ; indicate next subtest
007717' 260 17 0 00 007624*                     GO      ERESET                  ; do an EBUS reset
007720' 260 17 0 00 007705*                     GO      RDCSR                   ; read CSR register
007721' 255 00 0 00 000000                      JFCL                            ; error - ignore
007722' 603 01 0 00 000460                      TLNE    1,460                   ; bits all zero?
007723' 474 15 0 00 000000                      SETO    ERFLG,                  ; no - set error flag
007724' 027 00 0 00 007731'                     SCOPER  MB75                    ; print error message
007725' 254 00 0 00 007623'                     JRST    TA75                    ; loop on error
007726' 254 00 0 00 007730'                     JRST    TX75                    ; altmode exit
007727' 326 15 0 00 007730'                     JUMPN   ERFLG,TX75              ; error? yes - exit

                                        ; End of test

007730' 263 17 0 00 000000              TX75:   RTN                             ; return
```

```

                                  ; Error messages

007731' 270000  007732'           MB75:   LAST!CALL!TXALL!MB75PN          ; print data

007732' 200 02 0 00 007716*       MB75PN: MOVE    2,TSTSUB                ; get subtest number
007733' 256 00 0 02 007735'               XCT     MB75PP-1(2)             ; print text
007734' 260 17 0 00 010231'               GO      MPNT2                   ; print CSR data
007735' 263 17 0 00 000000                RTN

                                  MB75PP: TMSG    <
                                  Did EBUS Reset
007736' 037 00 0 00 013052'       CSR Bits 27 & 30-31 not zero>
                                          TMSG    <
                                  Did EBUS Reset, then wrote 0's to CSR
007737' 037 00 0 00 013064'       CSR Bits 27 & 30-31 not zero>
                                          TMSG    <
                                  Did EBUS Reset, then wrote 1's to CSR
007740' 037 00 0 00 013102'       CSR Bits 27 & 30-31 not set>
                                          TMSG    <
                                  Did EBUS Reset, then wrote 1's to CSR, then wrote 0's to CSR
007741' 037 00 0 00 013120'       CSR Bits 27 & 30-31 not still set>
                                          TMSG    <
                                  Wrote 1's to CSR with CSR bits 27 & 30-31 set
007742' 037 00 0 00 013144'       CSR Bits 27 & 30-31 not still set>
                                          TMSG    <
                                  Did EBUS Reset with CSR bits 27 & 30-31 set
007743' 037 00 0 00 013165'       CSR Bits 27 & 30-31 not cleared>
```

```

                                          ;#*****************************************************************
                                          ;* TEST 76 - CSR 27 & 30-31 Port-side
                                          ;
                                          ; Description:  Verify  that  CSR  Bits 27 and 30-31 can be read
                                          ;               and written properly from the port side.
                                          ;
                                          ; Procedure:    KL> Load test ucode
                                          ;               KL> EBUS Reset
                                          ;               KL> Start port at location 0
                                          ;
                                          ;               UC> Read CSR and verify that bits 27,30-31 are 0's
                                          ;                   If error - set completion code in EBUF of 1
                                          ;
                                          ;               UC> Write 0's to bits 27,30-31
                                          ;               UC> Read CSR and verify that bits 27,30-31 are 0's
                                          ;                   If error - set completion code in EBUF of 2
                                          ;
                                          ;               UC> Write 1's to bits 27,30-31
                                          ;               UC> Read CSR and verify that bits 27,30-31 are 0's
                                          ;                   If error - set completion code in EBUF of 3
                                          ;
                                          ;               KL> Start port at location 40
                                          ;               KL> Write 1's to bits 27,30-31
                                          ;               UC> Read CSR and verify that bits 27,30-31 are 1's
                                          ;                   If error - set completion code in EBUF of 4
                                          ;
                                          ;               UC> Write 1's to bits 27,30-31
                                          ;               UC> Read CSR and verify that bits 27,30-31 are 0's
                                          ;                   If error - set completion code in EBUF of 5
                                          ;
                                          ;               If no errors - set completion code in EBUF of 0
                                          ;
                                          ; Failure:      ---
                                          ;#*****************************************************************

                                          ; Test data

 007744'  254 00 0 00 007751'             TSTE76: JRST    TG76                            ; go start test
 007745'  400400  000076                          EBUS!NDMP!ZEBUS!76                      ; test mask
 007746'  010053' 013205'                         T76M,,[ASCIZ ^CSR 27 & 30-31 Port-side^]
 007747'  010761' 012563'                         [EXP MLAST!E11],,[EXP E9,E1,MLAST!E6]
 007750'  777777  777777                          -1                                      ; failure test table

                                          ; Start test

 007751'  201 00 0 00 000000'             TG76:   MOVEI   Z2                              ; get address of module start
 007752'  260 17 0 00 007622*                     GO      TRACE                           ; handle trace output
 007753'  201 01 0 00 010053'                     MOVEI   1,T76M                          ; set up microcode address
 007754'  260 17 0 00 007414*                     GO      TLOAD                           ; load/verify it
 007755'  263 17 0 00 000000                      RTN                                     ; failed - exit test
```

```

                                                ; KL> EBUS Reset
                                                ; KL> Start port at location 0

007756' 400 15 0 00 000000      TA76:   SETZ    ERFLG,                  ; clear error flag
007757' 260 17 0 00 007717*             GO      ERESET                  ; do an EBUS Reset
007760' 201 00 0 00 000001              MOVEI   1                       ; set subtest number
007761' 202 00 0 00 007732*             MOVEM   TSTSUB                  ;   to 1
007762' 201 00 0 00 000010              MOVEI   MPRUN                   ; get startup bits
007763' 202 00 0 00 007423*             MOVEM   SDATA                   ; save in startup data word
007764' 402 00 0 00 007424*             SETZM   SNEXT                   ; set start address to 0
007765' 260 17 0 00 010277'             GO      MSTART                  ; start up port
007766' 254 00 0 00 007756'             JRST    TA76                    ; loop on error
007767' 254 00 0 00 010036'             JRST    TX76                    ; altmode exit
007770' 326 15 0 00 010036'             JUMPN   ERFLG,TX76              ; error? yes - exit

                                ; Port does the remainder of the testing and returns completion code
                                ; in the EBUF.

007771' 350 00 0 00 007761*             AOS     TSTSUB                  ; set up subtest number
007772' 201 01 0 00 200000              MOVEI   1,TSTEBF                ; set up to read EBUF
007773' 260 17 0 00 007704*             GO      LDCSR                   ; stop the port
007774' 260 17 0 00 007434*             GO      RDEBUF                  ; read completion code
007775' 405 01 0 00 000007              ANDI    1,7                     ; mask out irrelevant bits
007776' 202 01 0 00 013665'             MOVEM   1,SAVEBU                ; save completion code
007777' 332 00 0 00 000001              SKIPE   1                       ; error code set?
010000' 474 15 0 00 000000              SETO    ERFLG,                  ; yes - set error flag
010001' 027 00 0 00 010037'             SCOPER  MB76                    ; print error message
010002' 254 00 0 00 007756'             JRST    TA76                    ; loop on error
010003' 254 00 0 00 010036'             JRST    TX76                    ; altmode exit
010004' 326 15 0 00 010036'             JUMPN   ERFLG,TX76              ; error? yes - exit

                                ; Start up the port for the remainder of the testing

010005' 350 01 0 00 007771*             AOS     1,TSTSUB                ; increment test segment
010006' 202 01 0 00 013674'             MOVEM   1,T76SAV#               ; save it
010007' 200 01 0 00 013674'     TB76:   MOVE    1,T76SAV                ; get test segment number
010010' 202 01 0 00 010005*             MOVEM   1,TSTSUB                ; set up test segment
010011' 260 17 0 00 007757*             GO      ERESET                  ; do an EBUS Reset
010012' 201 00 0 00 000470              MOVEI   MPRUN!460               ; get startup bits
010013' 202 00 0 00 007763*             MOVEM   SDATA                   ; save in startup data word
010014' 201 00 0 00 000040              MOVEI   40                      ; get start address 40
010015' 202 00 0 00 007764*             MOVEM   SNEXT                   ;   and set it up
010016' 260 17 0 00 010277'             GO      MSTART                  ; start up port
010017' 254 00 0 00 010007'             JRST    TB76                    ; loop on error
010020' 254 00 0 00 010036'             JRST    TX76                    ; altmode exit
010021' 326 15 0 00 010036'             JUMPN   ERFLG,TX76              ; error? yes - exit
```

```

                                     ; Port does the remainder of the testing and returns completion code
                                     ; in the EBUF.

010022' 350 00 0 00 010010*                   AOS     TSTSUB                  ; increment test segment
010023' 201 01 0 00 200000                    MOVEI   1,TSTEBF                ; set up to read EBUF
010024' 260 17 0 00 007773*                   GO      LDCSR                   ; stop the port
010025' 260 17 0 00 007774*                   GO      RDEBUF                  ; read completion code
010026' 405 01 0 00 000007                    ANDI    1,7                     ; mask out irrelevant bits
010027' 202 01 0 00 013665'                   MOVEM   1,SAVEBU                ; save completion code
010030' 332 00 0 00 000001                    SKIPE   1                       ; error code set?
010031' 474 15 0 00 000000                    SETO    ERFLG,                  ; yes - set error flag
010032' 027 00 0 00 010037'                   SCOPER  MB76                    ; print error message
010033' 254 00 0 00 010007'                   JRST    TB76                    ; loop on error
010034' 254 00 0 00 010036'                   JRST    TX76                    ; altmode exit
010035' 326 15 0 00 010036'                   JUMPN   ERFLG,TX76              ; error? yes - exit

                                     ; End of test

010036' 263 17 0 00 000000           TX76:    RTN                             ; return

                                     ; Error messages

010037' 270000  010040'              MB76:    LAST!CALL!TXALL!MB76PN          ; print data

010040' 200 02 0 00 013665'          MB76PN: MOVE     2,SAVEBU                ; get subtest number
010041' 256 00 0 02 010043'                  XCT      MB76PP-1(2)             ; print text
010042' 260 17 0 00 010231'                  GO       MPNT2                   ; print CSR data
010043' 263 17 0 00 000000                   RTN

                                     MB76PP: TMSG     <
                                     Did EBUS Reset, Port read CSR
010044' 037 00 0 00 013212'          CSR Bits 27 & 30-31 not zero>
                                             TMSG     <
                                     Did EBUS Reset, Port wrote 0's to CSR bits 27 & 30-31
010045' 037 00 0 00 013227'          CSR Bits 27 & 30-31 not zero>
                                             TMSG     <
                                     Did EBUS Reset, Port wrote 0's to CSR bits 27 & 30-31
                                     Then wrote 1's to CSR bits 27 & 30-31
010046' 037 00 0 00 013251'          CSR Bits 27 & 30-31 not zero>
                                             TMSG     <
                                     KL wrote 1's to CSR bits 27 & 30-31
010047' 037 00 0 00 013302'          Port read CSR, CSR Bits 27 & 30-31 not 1's>
                                             TMSG     <
                                     KL wrote 1's to CSR bits 27 & 30-31
                                     Port wrote 1's to CSR bits 27 & 30-31
010050' 037 00 0 00 013323'          Port read CSR, CSR Bits 27 & 30-31 not now zero>
010051' 037 00 0 00 013030'                  TMSGC    <Failure code incorrect - 6>
010052' 037 00 0 00 013036'                  TMSGC    <Failure code incorrect - 7>
```

```



                                  ; Microcode:    Build pattern in R2

010053' 000000  010023            T76M:   MWORD   <ADDR=0,JMAP,J=1,SDO,OR,B=2,D=2,SKCN,MGC=23>    ; 0
010054' 732001  240040
010055' 000100  310000                    MWORD   <LDCT,J=31>                                      ; 1
010056' 000000  000300
010057' 000200  020000                    MWORD   <RPCT,J=2,SOB,OR,B=2,D=5>                        ; 2
010060' 335001  000220
010061' 000300  030000                    MWORD   <LDCT,J=3>                                       ; 3
010062' 000000  000300
010063' 000400  040000                    MWORD   <RPCT,J=4,SOB,OR,B=2,D=7>                        ; 4
010064' 337001  000220

                                  ;       Read CSR and verify that bits 27,30-31 are 0's

010065' 000500  070040                    MWORD   <CJP,J=7,CENA,CCGC,SELE,MGC=40>                  ; 5
010066' 000400  015060
010067' 000600  050000                    MWORD   <JMAP,J=5>                                       ; 6
010070' 000000  000040
010071' 000700  000100                    MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>               ; 7
010072' 732000  405340
010073' 001000  000000                    MWORD   <CONT,SAB,A=2,B=1,AND,D=2>                       ; 10
010074' 142020  400340
010075' 001100  130000                    MWORD   <CJP,J=13,SAB,AND,A=1,B=2,D=1,CENA,CCFZ>         ; 11
010076' 141411  020060
010077' 001201  000001                    MWORD   <JMAP,J=100,SDO,OR,D=2,SKCN,MGC=1>               ; 12
010100' 732000  240040

                                  ;       UC> Write 0's to bits 27,30-31
                                  ;       UC> Read CSR and verify that bits 27,30-31 are 0's

010101' 001300  002004                    MWORD   <CONT,SOA,AND,D=1,OENA,SELE,MGC=4>               ; 13
010102' 441000  005340
010103' 001400  160040                    MWORD   <CJP,J=16,CENA,CCGC,SELE,MGC=40>                 ; 14
010104' 000400  015060
010105' 001500  140000                    MWORD   <JMAP,J=14>                                      ; 15
010106' 000000  000040
010107' 001600  000200                    MWORD   <CONT,SELE,MGC=200>                              ; 16
010110' 000000  005340

010111' 001700  210040                    MWORD   <CJP,J=21,CENA,CCGC,SELE,MGC=40>                 ; 17
010112' 000400  015060
010113' 002000  170000                    MWORD   <JMAP,J=17>                                      ; 20
010114' 000000  000040
010115' 002100  000100                    MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>               ; 21
010116' 732000  405340
010117' 002200  000000                    MWORD   <CONT,SAB,A=2,B=1,AND,D=2>                       ; 22
010120' 142020  400340
010121' 002300  250000                    MWORD   <CJP,J=25,SAB,AND,A=1,B=2,D=1,CENA,CCFZ>         ; 23
010122' 141411  020060
010123' 002401  000002                    MWORD   <JMAP,J=100,SDO,OR,D=2,SKCN,MGC=2>               ; 24
010124' 732000  240040

                                  ;       UC> Write 1's to bits 27,30-31
```

```
EBUSPE
ECOMP
ERESET
ERFLG
GENEPE
IADDR
ICALL
ICALL
ICSR
IDLE
IEXEC
IEXIT
IFLAG
IIOPF
IIPNT
IJRST
ILAST
INITP
INTNU
INTPN
INTPN
INTPN
INTTY
IPACL
IPASR
ISETD
ISETE
ISETE
ISETI
ISETI
ISTAR
```

```
 7433                                   ;       UC> Read CSR and verify that bits 27,30-31 are 0's
 7434
 7435   010125' 002500  002004                  MWORD   <CONT,SOA,OR,A=2,D=1,OENA,SELE,MGC=4>           ; 25
 7436   010126' 431020  005340
 7437   010127' 002600  300040                  MWORD   <CJP,J=30,CENA,CCGC,SELE,MGC=40>                ; 26
 7438   010130' 000400  015060
 7439   010131' 002700  260000                  MWORD   <JMAP,J=26>                                     ; 27
 7440   010132' 000000  000040
 7441   010133' 003000  000200                  MWORD   <CONT,SELE,MGC=200>                             ; 30
 7442   010134' 000000  005340
 7443
 7444   010135' 003100  330040                  MWORD   <CJP,J=33,CENA,CCGC,SELE,MGC=40>                ; 31
 7445   010136' 000400  015060
 7446   010137' 003200  310000                  MWORD   <JMAP,J=31>                                     ; 32
 7447   010140' 000000  000040
 7448   010141' 003300  000100                  MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>              ; 33
 7449   010142' 732000  405340
 7450   010143' 003400  000000                  MWORD   <CONT,SAB,A=2,B=1,AND,D=2>                      ; 34
 7451   010144' 142020  400340
 7452   010145' 003500  370000                  MWORD   <CJP,J=37,SAB,AND,A=1,B=2,D=1,CENA,CCFZ>        ; 35
 7453   010146' 141411  020060
 7454   010147' 003601  000003                  MWORD   <JMAP,J=100,SDO,OR,D=2,SKCN,MGC=3>              ; 36
 7455   010150' 732000  240040
 7456   010151' 003701  000000                  MWORD   <JMAP,J=100,SDO,OR,D=2,SKCN,MGC=0>              ; 37
 7457   010152' 732000  240040
 7458
 7459                                   ;       UC> Read CSR and verify that bits 27,30-31 are 1's
 7460
 7461   010153' 004000  410000                  MWORD   <ADDR=40,JMAP,J=41>                             ; 40
 7462   010154' 000000  000040
 7463   010155' 004100  430040                  MWORD   <CJP,J=43,CENA,CCGC,SELE,MGC=40>                ; 41
 7464   010156' 000400  015060
 7465   010157' 004200  410000                  MWORD   <JMAP,J=41>                                     ; 42
 7466   010160' 000000  000040
 7467   010161' 004300  000100                  MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>              ; 43
 7468   010162' 732000  405340
 7469   010163' 004400  000000                  MWORD   <CONT,SAB,A=2,B=1,AND,D=2>                      ; 44
 7470   010164' 142020  400340
 7471   010165' 004500  470000                  MWORD   <CJP,J=47,SAB,XOR,A=1,B=2,D=1,CENA,CCFZ>        ; 45
 7472   010166' 161411  020060
 7473   010167' 004601  000004                  MWORD   <JMAP,J=100,SDO,OR,D=2,SKCN,MGC=4>              ; 46
 7474   010170' 732000  240040
 7475
 7476                                   ;       UC> Write 1's to bits 27,30-31
 7477                                   ;       UC> Read CSR and verify that bits 27,30-31 are 0's
 7478
 7479   010171' 004700  002004                  MWORD   <CONT,SOA,OR,A=2,D=1,OENA,SELE,MGC=4>           ; 47
 7480   010172' 431020  005340
 7481   010173' 005000  520040                  MWORD   <CJP,J=52,CENA,CCGC,SELE,MGC=40>                ; 50
 7482   010174' 000400  015060
 7483   010175' 005100  500000                  MWORD   <JMAP,J=50>                                     ; 51
 7484   010176' 000000  000040
 7485   010177' 005200  000200                  MWORD   <CONT,SELE,MGC=200>                             ; 52
 7486   010200' 000000  005340
 7487
```

0627

```
  7488  010201' 005300  550040                      MWORD   <CJP,J=55,CENA,CCGC,SELE,MGC=40>            ; 53
  7489  010202' 000400  015060
  7490  010203' 005400  530000                      MWORD   <JMAP,J=53>                                 ; 54
  7491  010204' 000000  000040
  7492  010205' 005500  000100                      MWORD   <CONT,SDO,B=1,OR,D=2,SELE,MGC=100>          ; 55
  7493  010206' 732000  405340
  7494  010207' 005600  000000                      MWORD   <CONT,SAB,A=2,B=1,AND,D=2>                  ; 56
  7495  010210' 142020  400340
  7496  010211' 005700  610000                      MWORD   <CJP,J=61,SAB,AND,A=1,B=2,D=1,CENA,CCFZ>    ; 57
  7497  010212' 141411  020060
  7498  010213' 006001  000005                      MWORD   <JMAP,J=100,SDO,OR,D=2,SKCN,MGC=5>          ; 60
  7499  010214' 732000  240040
  7500  010215' 006101  000000                      MWORD   <JMAP,J=100,SDO,OR,D=2,SKCN,MGC=0>          ; 61
  7501  010216' 732000  240040
  7502
  7503                                      ;       UC> Write completion code
  7504
  7505  010217' 010000  002004                      MWORD   <ADDR=100,CONT,SOA,OR,D=1,OENA,SELE,MGC=4>      ; 100
  7506  010220' 431000  005340
  7507  010221' 010101  010000                      MWORD   <JMAP,J=101,D=1>                                ; 101
  7508  010222' 001000  000040
  7509  010223' 777777  777777                      -1
  7510
```

```
MA5
MA54
MA54PN
MA5PNT
MA6
MA60
MA60PN
MA64
MA65
MA7
MA70
MA71
MA72
MA7PNT
MB14
MB22
MB25
MB26
MB27
MB30
MB31
MB32
MB33
MB34
MB42
MB43
MB44
MB45
MB46
MB47
MB50
MB51
MB53
MB55
MB56
MB57
MB61
MB62
MB62P
MB63
MB7
MB73
MB73P
MB73P
MB74
MB74P
MB74P
MB75
MB75P
MB75P
MB76
MB76P
MB76P
```

0628

MB7PNT

```
                                                SUBTTL  EBUS Module Error Printing Routines

                                        ;#**********************************************************************
                                        ; MPNT1 - Print Port/RH Interaction Message
                                        ;#**********************************************************************

010224' 037 00 0 00 013355'             MPNT1:  TMSGC   <Port/RH>
010225' 135 00 0 00 010471'                     LDB     [POINT 3,MBCN,9]        ; get basic 3 bits
010226' 037 16 0 00 000003                      PNTOCS
010227' 037 00 0 00 013357'                     TMSG    < Interaction error>
010230' 263 17 0 00 000000                      RTN

                                        ;#**********************************************************************
                                        ; MPNT2 - Print CSR Contents (Prints contents of SAVCSR)
                                        ;#**********************************************************************

010231' 200 01 0 00 010365'             MPNT2:  MOVE    1,SAVCSR                ; get CSR data
010232' 201 02 0 00 000023                      MOVEI   2,^D19                  ; character count so far
010233' 201 03 0 00 000023                      MOVEI   3,^D19                  ; continuation line column number
010234' 260 17 0 00 000000*                     GO      CSRPNT                  ; go print in English
010235' 263 17 0 00 000000                      RTN


                                        ;#**********************************************************************
                                        ; MPNT4 - Print EBUF data correct/actual (data in SAVEB1,SAVEB2)
                                        ;#**********************************************************************

010236' 200 01 0 00 003433*             MPNT4:  MOVE    1,SCOSW                 ; get switches
010237' 603 01 0 00 000200                      TLNE    1,TXTINH                ; inhibit text?
010240' 037 00 0 00 013363'                     TMSGC   <EBUF (C):  >           ; yes
010241' 607 01 0 00 000200                      TLNN    1,TXTINH                ; inhibit text?
010242' 037 00 0 00 013366'                     TMSGC   <EBUF (Correct):  >     ; no
010243' 200 00 0 00 010372'                     MOVE    SAVEB1                  ; get EBUF data written
010244' 037 13 0 00 000000                      PNTHW
010245' 603 01 0 00 000200                      TLNE    1,TXTINH                ; inhibit text?
010246' 037 00 0 00 013372'                     TMSGC   <     (A):  >           ; yes
010247' 607 01 0 00 000200                      TLNN    1,TXTINH                ; inhibit text?
010250' 037 00 0 00 013375'                     TMSGC   <     (Actual):  >      ; no
010251' 200 00 0 00 010373'                     MOVE    SAVEB2                  ; get EBUF data
010252' 037 13 0 00 000000                      PNTHW
010253' 263 17 0 00 000000                      RTN
```

```
MB7PNT
MBCN

MBERR
MBUS
MC47
MC50
MC51
MC63
MD47
MD50
MD51
MLAST

MPNT1
MPNT2

MPNT4
MPNT5
MPNT6
MPRUN

MSG

MSTARE
MSTART

NDMP

ODELA
P
PAT
PCLEAF
PF500
PF501
PF502
PFAIL
PFAPR
PFPNT
PISYOI
PISYON
```

DFPTA2 MACRO %53B(1242) 07:48 1-Jul-85 Page 150

RDCSR

```


                                        ;#*******************************************************************
                                        ; MPNT5 - Print CSR data correct/actual (data in SAVCS2,SAVCS3)
                                        ;#*******************************************************************

010254' 200 01 0 00 010236*     MPNT5:  MOVE    1,SCOSW                 ; get switches
010255' 603 01 0 00 000200              TLNE    1,TXTINH                ; inhibit text?
010256' 037 00 0 00 013401'             TMSGC   <CSR (C):  >            ; yes
010257' 607 01 0 00 000200              TLNN    1,TXTINH                ; inhibit text?
010260' 037 00 0 00 013404'             TMSGC   <CSR (Correct):  >      ; no
010261' 200 00 0 00 010367'             MOVE    SAVCS2                  ; get CSR data written
010262' 037 13 0 00 000000              PNTHW
010263' 603 01 0 00 000200              TLNE    1,TXTINH                ; inhibit text?
010264' 037 00 0 00 013410'             TMSGC   <    (A):  >            ; yes
010265' 607 01 0 00 000200              TLNN    1,TXTINH                ; inhibit text?
010266' 037 00 0 00 013413'             TMSGC   <     (Actual):  >      ; no
010267' 200 00 0 00 010370'             MOVE    SAVCS3                  ; get CSR data
010270' 037 13 0 00 000000              PNTHW
010271' 263 17 0 00 000000              RTN


                                        ;#*******************************************************************
                                        ; MPNT6 - Read and print CSR
                                        ;#*******************************************************************

010272' 260 17 0 00 007720*     MPNT6:  GO      RDCSR                   ; get CSR contents
010273' 255 00 0 00 000000              JFCL                            ; error
010274' 202 01 0 00 010365'             MOVEM   1,SAVCSR                ; save it
010275' 260 17 0 00 010231'             GO      MPNT2                   ; print it
010276' 263 17 0 00 000000              RTN                             ; return
```

0629

```
RDCSR

RDEBUF
RQINT
SAVCS1
SAVCS2

SAVCS3
SAVCS4
SAVCSR

SAVDA1

SAVDA2
SAVDA3
SAVDAT
SAVEB1
SAVEB2
SAVEBL
SAVMBC

SCOSW
SDATA

SETVEC
SINCYC
SNEXT

SSLAST
SSSTRT

T13M
T16PAT
T22M
T23PAT
T25M
T26M
T27M
T30M
T31M
T32M
T33M
T34M
T35M
T36M
```

DFPTA2 MACRO %53B(1242) 07:48 1-Jul-85 Page 151

```

                                            ;#*******************************************************************
                                            ; MSTART - Start up the port and handle errors if fails to start
                                            ;#*******************************************************************

        010277' 260 17 0 00 007005*         MSTART: GO      IPACLR                  ; do a 'port clear'
        010300' 260 17 0 00 000000*                 GO      IPASRT                  ; start up port
        010301' 474 15 0 00 000000                  SETO    ERFLG,                  ; failed
        010302' 474 15 0 00 000000                  SETO    ERFLG,                  ; failed
        010303' 474 15 0 00 000000                  SETO    ERFLG,                  ; failed
        010304' 027 00 0 00 010312'                 SCOPER  MSTARE                  ; print error message
        010305' 263 17 0 00 000000                  RTN                             ; loop on error (return +1)
                                                    PJRST   [AOS    (P)             ; altmode exit (return +2)
        010306' 254 00 0 00 013417'                          RTN]
        010307' 350 00 0 17 000000                  AOS     (P)                     ; ok (return +3)
        010310' 350 00 0 17 000000                  AOS     (P)                     ; ok (return +3)
        010311' 263 17 0 00 000000                  RTN                             ; return

                                            ; Error message - Couldn't start port

        010312' 160000  011773'             MSTARE: MSG!TXALL![ASCIZ /Failed to start port properly/]
        010313' 270000  010272'                     LAST!CALL!TXALL!MPNT6


                                            ;#*******************************************************************
                                            ;* PFPNT - Print page fail data
                                            ;#*******************************************************************

        010314' 037 00 0 00 013421'         PFPNT:  TMSGC   <APR status:  >
        010315' 200 00 0 00 000000*                 MOVE    PFAPR                   ; get APR status
        010316' 037 13 0 00 000000                  PNTHW                           ; print it
        010317' 037 00 0 00 013425'                 TMSGC   <EPT500/ >              ; print EPT location 500
        010320' 200 00 0 00 000000*                 MOVE    PF500                   ; get data
        010321' 037 13 0 00 000000                  PNTHW                           ; print it
        010322' 037 00 0 00 013430'                 TMSGC   <EPT501/ >              ; print EPT location 501
        010323' 200 00 0 00 000000*                 MOVE    PF501                   ; get data
        010324' 037 13 0 00 000000                  PNTHW                           ; print it
        010325' 037 00 0 00 013433'                 TMSGC   <EPT502/ >              ; print EPT location 502
        010326' 200 00 0 00 000000*                 MOVE    PF502                   ; get data
        010327' 037 13 0 00 000000                  PNTHW                           ; print it
        010330' 263 17 0 00 000000                  RTN
```

```
T40DAT
T40M
T41DAT
T41M
T42M
T43M
T44M
T45M
T46M
T47M
T50M
T51M
T53M
T55M
T56M
T57M
T60M
T61M
T62M
T63M
T64ADD
T64ADI
T64ADR
T64ADT
T64M
T65ADD
T65ADI
T65ADR
T65ADT
T65M
T70ADD
T70ADI
T70ADR
T70ADT
T70M
T71ADD
T71ADI
T71ADR
T71ADT
T71M
T72ADD
T72ADI
T72ADR
T72ADT
T72M
T73M
T74M
T76M
T76SAV
TA1
TA10
TA11
TA12
```

```

                                       ;#**********************************************************************
                                       ;* INTPNT - Print interrupt activity
                                       ;#**********************************************************************

010331' 261 17 0 00 000000             INTPNT: RPUT    (0,1,2)                 ; save AC's
010334' 037 00 0 00 013436'                    TMSGC   <Interrupt history:  (>
010335' 200 00 0 00 005542*                    MOVE    INTNUM                  ; get number of interrupts
010336' 037 15 0 00 000000                     PNTDEC
010337' 037 00 0 00 013443'                    TMSG    <. interrupts occurred)>

                                       ; Print them

010340' 335 00 0 00 010335*            INTPNO: SKIPGE  INTNUM                  ; any interrupt left?
010341' 254 00 0 00 010355'                    JRST    INTPNX                  ; no - exit
010342' 370 01 0 00 010340*                    SOS     1,INTNUM                ; point to previous interrupt
010343' 303 01 0 00 000012                     CAILE   1,^D10                  ; overflow interrupt?
010344' 201 01 0 00 000012                     MOVEI   1,^D10                  ; yes - limit to 10 interrupts
010345' 202 01 0 00 010342*                    MOVEM   1,INTNUM                ; save interrupt number
010346' 037 00 0 00 013450'                    TMSGC   <  #>
010347' 200 00 0 00 000001                     MOVE    1
010350' 037 15 0 00 000000                     PNTDEC
010351' 037 00 0 00 013452'                    TMSG    <. >
010352' 550 02 0 01 005544*                    HRRZ    2,INTTYP(1)             ; get type
                                               XCT     [TMSG   <Vectored>      ; print it
                                                        TMSG   <NonVect Chn 1>
                                                        TMSG   <NonVect Chn 2>
                                                        TMSG   <NonVect Chn 3>
                                                        TMSG   <NonVect Chn 4>
                                                        TMSG   <NonVect Chn 5>
                                                        TMSG   <NonVect Chn 6>
010353' 256 00 0 02 013502'                             TMSG   <NonVect Chn 7>](2)
010354' 254 00 0 00 010340'                    JRST    INTPNO                  ; loop till done

                                       ; Exit

010355' 262 17 0 00 000002             INTPNX: RGET    (2,1,0)                 ; yes - restore AC's
010360' 263 17 0 00 000000                     RTN                             ; exit
```

```
TA13
TA14
TA15
TA16
TA17
TA2
TA20
TA21
TA22
TA23
TA24
TA25
TA26
TA27
TA2L
TA2X
TA3
TA30
TA31
TA32
TA33
TA34
TA35
TA36
TA37
TA3L
TA3X
TA4
TA40
TA41
TA42
TA43
TA44
TA45
TA46
TA47
TA4L
TA4X
TA5
TA50
TA51
TA53
TA54
TA55
TA56
TA57
TA5L
TA5X
TA6
TA61
TA62
TA63
TA64
```

0632

```

                                     ;#***********************************************************************
                                     ; Test temporary storage
                                     ;#***********************************************************************

     010361' 000000  000000          SAVDAT: 0                                 ; location for misc data
     010362' 000000  000000          SAVDA1: 0                                 ; location for misc data
     010363' 000000  000000          SAVDA2: 0                                 ; location for misc data
     010364' 000000  000000          SAVDA3: 0                                 ; location for misc data
     010365' 000000  000000          SAVCSR: 0                                 ; location for CSR data
     010366' 000000  000000          SAVCS1: 0                                 ; location for CSR data written
     010367' 000000  000000          SAVCS2: 0                                 ; location for CSR data correct
     010370' 000000  000000          SAVCS3: 0                                 ; location for CSR data actual
     010371' 000000  000000          SAVCS4: 0                                 ; location for misc CSR data
     010372' 000000  000000          SAVEB1: 0                                 ; location for EBUF data correct
     010373' 000000  000000          SAVEB2: 0                                 ; location for EBUF data actual

                                     ;#***********************************************************************
                                     ; End of EBUS Module Tests
                                     ;#***********************************************************************

                                             XLIST


NO ERRORS DETECTED

PROGRAM BREAK IS 013675
CPU TIME USED 04:06.050

167P CORE USED
```

```
TA65
TA7
TA70
TA71
TA72
TA73
TA74
TA75
TA76
TA7PAT
TB10
TB13
TB22
TB27
TB42
TB43
TB44
TB45
TB46
TB47
TB50
TB51
TB53
TB54
TB55
TB56
TB57
TB61
TB62
TB62A
TB63
TB64
TB65
TB70
TB71
TB72
TB76
TC47
TC50
TC51
TC63
TD47
TD50
TD51
TG1
TG10
TG11
TG12
TG13
TG14
TG15
TG16
TG17
```

```
0633
```

```
DFPTA2  MACRO %53B(1242) 07:48  1-Jul-85 Page S-1
DFPTA2  MAC     18-Apr-85 10:58         SYMBOL TABLE                                                               SEQ 0646

AEBUF            002700' ext  IJRST            000011  spd  MA41PN           003433'      MB76PN           010040'
BPPNT            000000  ext  ILAST            000000  spd  MA46             004024'      MB76PP           010044'
BEXEC            003162' ext  INITPI           005522' ext  MA4PNA           000271'      MB7PNT           000523'
BUFF             012555' ext  INTNUM           010345' ext  MA4PNB           000275'      MBCN             000016
BUFGEN           007001' ext  INTPNO           010340'      MA5              000352'      MBERR    002000  000000  spd
CALL     200000  000000  spd  INTPNT           010331'      MA54             004744'      MBUS     010000  000000  spd
CEBUF            002676' ext  INTPNX           010355'      MA54PN           004746'      MC47             004126'
CRAMPE   004000  000000  spd  INTTYP           010352' ext  MA5PNT           000354'      MC50             004325'
CSRCHN   100000  000000  spd  IPACLR           010277' ext  MA6              000430'      MC51             004523'
CSRENG           000000  ext  IPASRT           010300' ext  MA60             005223'      MC63             005572'
CSRPNT           010234' ext  ISETDE           000002  spd  MA60PN           005224'      MD47             004130'
CSRRQS   200000  000000  spd  ISETEI           000005  spd  MA64             005701'      MD50             004327'
DCOMP    000040  000000  spd  ISETEX           000001  spd  MA65             006125'      MD51             004525'
E1               000001  spd  ISETID           000003  spd  MA7              000512'      MLAST    400000  000000  spd
E10              000012  spd  ISETIN           000004  spd  MA70             006423'      MPNT1            010224'
E11              000013  spd  ISTART           000012  spd  MA71             006657'      MPNT2            010231'
E12              000014  spd  IWAIT            000010  spd  MA72             007120'      MPNT4            010236'
E13              000015  spd  LAST     010000  000000  spd  MA7PNT           000514'      MPNT5            010254'
E14              000016  spd  LDCSR            010024' ext  MB14             001157'      MPNT6            010272'
E15              000017  spd  LDEBUF           002016' ext  MB22             001557'      MPRUN            000010  spd
E16              000020  spd  LDRAR            007331' ext  MB25             002104'      MSG      100000  000000  spd
E17              000021  spd  M1               000031  spd  MB26             002200'      MSTARE           010312'
E18              000022  spd  M11              000043  spd  MB27             002257'      MSTART           010277'
E19              000023  spd  M12              000044  spd  MB30             002334'      NDMP     000400  000000  spd
E2               000002  spd  M3               000033  spd  MB31             002410'      ODELAY           005356' ext
E20              000024  spd  MA1              000021'      MB32             002470'      P                000017
E21              000025  spd  MA10             000675'      MB33             002543'      PAT              000014
E22              000026  spd  MA10PN           000677'      MB34             002631'      PCLEAR           400000  spd
E23              000027  spd  MA11             000733'      MB42             003561'      PF500            010320' ext
E24              000030  spd  MA12             000770'      MB43             003632'      PF501            010323' ext
E3               000003  spd  MA13             001046'      MB44             003706'      PF502            010326' ext
E4               000004  spd  MA13PN           001050'      MB45             003751'      PFAIL            004736' ext
E5               000005  spd  MA14             001154'      MB46             004026'      PFAPR            010315' ext
E6               000006  spd  MA15             001216'      MB47             004124'      PFPNT            010314'
E7               000007  spd  MA16             001257'      MB50             004323'      PISYOF           005167' ext
E8               000010  spd  MA17             001422'      MB51             004521'      PISYON           005207' ext
E9               000011  spd  MA2              000075'      MB53             004677'      PJRST    254000  000000
EBUS     400000  000000  spd  MA20             001451'      MB55             005014'      PNT1     037040  000000
EBUSPE           004000  spd  MA21             001504'      MB56             005071'      PNT6     037300  000000
ECOMP    000020  000000  spd  MA23             001625'      MB57             005136'      PNTDEC   037640  000000
ERESET           010011' ext  MA23PN           001627'      MB61             005367'      PNTHW    037540  000000
ERFLG            000015       MA24             002033'      MB62             005465'      PNTMSG   037000  000000
GENEPE           100000  spd  MA24PN           002035'      MB62PN           005470'      PNTOCS   037700  000003
GET      262740  000000       MA2PNA           000102'      MB63             005567'      PUT      261740  000000
GO       260740  000000       MA2PNB           000106'      MB7              000521'      RDCSR            010272' ext
IADDR            007054' ext  MA3              000170'      MB73             007346'      RDEBUF           010025' ext
ICALL            000006  spd  MA34             002627'      MB73PN           007347'      RQINT    010000  000000  spd
ICALLC           000007  spd  MA35             002772'      MB73PP           007353'      RTN      263740  000000
ICSR             000000  ext  MA36             003177'      MB74             007446'      SAVCS1           010366'
IDLE     000100  000000  spd  MA37             003237'      MB74PN           007447'      SAVCS2           010367'
IEXEC            007006' ext  MA3PNT           000172'      MB74PP           007453'      SAVCS3           010370'
IEXIT            000013  spd  MA4              000264'      MB75             007731'      SAVCS4           010371'
IFLAG            000000  ext  MA40             003316'      MB75PN           007732'      SAVCSR           010365'
IIOPF            007067' ext  MA40PN           003320'      MB75PP           007736'      SAVDA1           010362'
IIPNT            000000  ext  MA41             003431'      MB76             010037'      SAVDA2           010363'
```

```
TG2
TG20
TG21
TG22
TG23
TG24
TG25
TG26
TG27
TG3
TG30
TG31
TG32
TG33
TG34
TG35
TG36
TG37
TG4
TG40
TG41
TG42
TG43
TG44
TG45
TG46
TG47
TG5
TG50
TG51
TG53
TG54
TG55
TG56
TG57
TG6
TG60
TG61
TG62
TG63
TG64
TG65
TG7
TG70
TG71
TG72
TG73
TG74
TG75
TG76
TL35
TL36
TL60
```

```
SAVDA3          010364'        T65ADR    013670'   TA41      003366'   TB64      005624'
SAVDAT          010361'        T65ADT    006110'   TA42      003534'   TB65      006050'
SAVEB1          010372'        T65M      006127'   TA43      003606'   TB70      006312'
SAVEB2          010373'        T70ADD    006352'   TA44      003662'   TB71      006545'
SAVEBU          013665'        T70ADI    006346'   TA45      003725'   TB72      007006'
SAVMBC          013666'        T70ADR    013671'   TA46      004000'   TB76      010007'
SCOPER   027000 000000         T70ADT    006417'   TA47      004051'   TC47      004075'
SCOSW           010254' ext    T70M      006425'   TA4L      000231'   TC50      004274'
SDATA           010013' ext    T71ADD    006605'   TA4X      000257'   TC51      004472'
SETVEC          005523' ext    T71ADI    006601'   TA5       000323'   TC63      005555'
SINCYC          020000  spd    T71ADR    013672'   TA50      004250'   TD47      004111'
SNEXT           010015' ext    T71ADT    006653'   TA51      004446'   TD50      004310'
SSLAST          000000  spd    T71M      006661'   TA53      004653'   TD51      004506'
SSSTRT          000001  spd    T72ADD    007040'   TA54      004715'   TG1       000005'
T13M            001063'        T72ADI    007036'   TA55      004770'   TG10      000627'
T16PAT          001261'        T72ADR    013673'   TA56      005043'   TG11      000715'
T22M            001561'        T72ADT    007110'   TA57      005110'   TG12      000745'
T23PAT          001634'        T72M      007122'   TA5L      000324'   TG13      001000'
T25M            002106'        T73M      007361'   TA5X      000345'   TG14      001115'
T26M            002202'        T74M      007462'   TA6       000416'   TG15      001170'
T27M            002261'        T76M      010053'   TA61      005336'   TG16      001225'
T30M            002336'        T76SAV    013674'   TA62      005422'   TG17      001405'
T31M            002412'        TA1       000007'   TA63      005521'   TG2       000035'
T32M            002472'        TA10      000631'   TA64      005623'   TG20      001432'
T33M            002545'        TA11      000717'   TA65      006047'   TG21      001460'
T34M            002633'        TA12      000751'   TA7       000443'   TG22      001515'
T35M            002775'        TA13      001007'   TA70      006311'   TG23      001572'
T36M            003201'        TA14      001122'   TA71      006544'   TG24      002000'
T40DAT          003312'        TA15      001175'   TA72      007005'   TG25      002050'
T40M            003336'        TA16      001230'   TA73      007234'   TG26      002145'
T41DAT          003425'        TA17      001407'   TA74      007416'   TG27      002222'
T41M            003451'        TA2       000042'   TA75      007623'   TG3       000132'
T42M            003563'        TA20      001436'   TA76      007756'   TG30      002302'
T43M            003634'        TA21      001463'   TA7PAT    000530'   TG31      002356'
T44M            003710'        TA22      001522'   TB10      000653'   TG32      002435'
T45M            003753'        TA23      001575'   TB13      001022'   TG33      002506'
T46M            004030'        TA24      002003'   TB22      001542'   TG34      002561'
T47M            004132'        TA25      002055'   TB27      002244'   TG35      002661'
T50M            004331'        TA26      002152'   TB42      003546'   TG36      003152'
T51M            004527'        TA27      002227'   TB43      003620'   TG37      003217'
T53M            004701'        TA2L      000043'   TB44      003674'   TG4       000222'
T55M            005016'        TA2X      000070'   TB45      003737'   TG40      003247'
T56M            005073'        TA3       000141'   TB46      004012'   TG41      003360'
T57M            005140'        TA30      002307'   TB47      004063'   TG42      003527'
T60M            005233'        TA31      002363'   TB50      004262'   TG43      003601'
T61M            005372'        TA32      002442'   TB51      004460'   TG44      003655'
T62M            005474'        TA33      002515'   TB53      004665'   TG45      003720'
T63M            005574'        TA34      002566'   TB54      004737'   TG46      003773'
T64ADD          005654'        TA35      002672'   TB55      005002'   TG47      004044'
T64ADI          005652'        TA36      003162'   TB56      005055'   TG5       000314'
T64ADR          013667'        TA37      003221'   TB57      005122'   TG50      004243'
T64ADT          005664'        TA3L      000142'   TB61      005354'   TG51      004441'
T64M            005703'        TA3X      000163'   TB62      005442'   TG53      004646'
T65ADD          006100'        TA4       000230'   TB62A     005444'   TG54      004711'
T65ADI          006076'        TA40      003255'   TB63      005540'   TG55      004763'
```

```
TG56           005036'        TSTA6          000000  ext    TSTE67         006261' ent    TX46           004023'
TG57           005103'        TSTA7          000000  ext    TSTE7          000432' ent    TX47           004123'
TG6            000410'        TSTE1          000000' ent    TSTE70         006266' ent    TX5            000350'
TG60           005150'        TSTE10         000616' ent    TSTE71         006522' ent    TX50           004322'
TG61           005327'        TSTE11         000704' ent    TSTE72         006764' ent    TX51           004520'
TG62           005411'        TSTE12         000735' ent    TSTE73         007217' ent    TX53           004676'
TG63           005512'        TSTE13         000773' ent    TSTE74         007402' ent    TX54           004743'
TG64           005611'        TSTE14         001106' ent    TSTE75         007613' ent    TX55           005013'
TG65           006035'        TSTE15         001162' ent    TSTE76         007744' ent    TX56           005070'
TG7            000440'        TSTE16         001220' ent    TSTEBF         200000  spd    TX57           005135'
TG70           006275'        TSTE17         001376' ent    TSTFLG         010514' ext    TX6            000427'
TG71           006530'        TSTE2          000025' ent    TSTPC          000000  ext    TX60           005222'
TG72           006771'        TSTE20         001424' ent    TSTSUB         010022' ext    TX61           005366'
TG73           007227'        TSTE21         001453' ent    TSTU10         000000  ext    TX62           005464'
TG74           007411'        TSTE22         001506' ent    TSTU11         000000  ext    TX63           005566'
TG75           007621'        TSTE23         001564' ent    TSTU12         000000  ext    TX64           005634'
TG76           007751'        TSTE24         001773' ent    TSTU13         000000  ext    TX65           006060'
TL35           002666'        TSTE25         002043' ent    TSTU41         000000  ext    TX7            000511'
TL36           003157'        TSTE26         002133' ent    TSTU6          000000  ext    TX70           006322'
TL60           005157'        TSTE27         002211' ent    TSTU7          000000  ext    TX71           006555'
TL64           005620'        TSTE3          000123' ent    TUSER   001000 000000  spd    TX72           007016'
TL65           006044'        TSTE30         002272' ent    TX1            000020'        TX73           007345'
TL70           006306'        TSTE31         002347' ent    TX10           000674'        TX74           007445'
TL71           006541'        TSTE32         002427' ent    TX11           000732'        TX75           007730'
TL72           007002'        TSTE33         002501' ent    TX12           000767'        TX76           010036'
TLOAD          007754' ext    TSTE34         002554' ent    TX13           001045'        TXALL   060000 000000  spd
TRACE          007752' ext    TSTE35         002644' ent    TX14           001153'        TXNOT   040000 000000  spd
TS35           002711'        TSTE36         003136' ent    TX15           001215'        TXTINH         000200  spd
TS36           003174'        TSTE37         003212' ent    TX16           001256'        TXYES   020000 000000  spd
TS64           005635'        TSTE4          000214' ent    TX17           001421'        UDEBUG         010513' ext
TS64A          005641'        TSTE40         003241' ent    TX2            000073'        USER           030037
TS64Z          005650'        TSTE41         003353' ent    TX20           001450'        Z2             000000'
TS65           006061'        TSTE42         003520' ent    TX21           001503'        ZEBUS          000000  spd
TS65A          006065'        TSTE43         003572' ent    TX22           001556'        $ARG2          000001
TS65Z          006074'        TSTE44         003643' ent    TX23           001624'        $B             000044
TS70           006323'        TSTE45         003713' ent    TX24           002032'        $CHR           000044
TS70A          006333'        TSTE46         003762' ent    TX25           002103'        $GARG          000001
TS70AA         006344'        TSTE47         004033' ent    TX26           002177'        %ADDR          000102  spd
TS70EX         006371'        TSTE5          000307' ent    TX27           002256'        %ML     010101 010000  spd
TS70Z          006342'        TSTE50         004233' ent    TX3            000166'        %MR     001000 000040  spd
TS71           006556'        TSTE51         004432' ent    TX30           002333'        .CONI          000420' ext
TS71A          006566'        TSTE52         004630' ent    TX31           002407'        .CONO          000243' ext
TS71AA         006577'        TSTE53         004635' ent    TX32           002467'        .DATAI         001033' ext
TS71DE         006626'        TSTE54         004704' ent    TX33           002542'        .DATAO         001032' ext
TS71Z          006575'        TSTE55         004751' ent    TX34           002626'        .LA            000000  spd
TS72           007017'        TSTE56         005025' ent    TX35           002710'        .LADDR  000100 000000  spd
TS72A          007027'        TSTE57         005076' ent    TX36           003173'        .LAND          000000  spd
TS72AA         007116'        TSTE6          000376' ent    TX37           003236'        .LB            000000  spd
TS72EX         007062'        TSTE60         005143' ent    TX4            000262'        .LBAD          000000  spd
TS72Z          007114'        TSTE61         005320' ent    TX40           003311'        .LCCCC         000000  spd
TSTA1          000000  ext    TSTE62         005401' ent    TX41           003424'        .LCCER         000000  spd
TSTA2          000000  ext    TSTE63         005503' ent    TX42           003560'        .LCCFZ         000000  spd
TSTA3          000000  ext    TSTE64         005603' ent    TX43           003631'        .LCCGC         000000  spd
TSTA4          000000  ext    TSTE65         006030' ent    TX44           003705'        .LCCIA         000000  spd
TSTA5          000000  ext    TSTE66         006254' ent    TX45           003750'        .LCENA         000000  spd
```

```
.LCJP               000000  spd      .MSDO               000007  spd
.LCJS               000000  spd      .MSELE              000007  spd
.LCONT              000000  spd      .MSKCN              000037  spd
.LCRTN              000000  spd      .MXNOR              000007  spd
.LCRY               000000  spd      .MXOR               000007  spd
.LD                 000000  spd      .PION               005524' ext
.LJ                 010000  spd      .RA       000010    000000  spd
.LJMAP              000000  spd      .RAND     040000    000000  spd
.LJZ                000000  spd      .RB                 400000  spd
.LLDCT              000000  spd      .RBAD               000001  spd
.LLDLM              000000  spd      .RCCCC              030000  spd
.LMGC               000001  spd      .RCCER              100000  spd
.LOENA              002000  spd      .RCCFZ              020000  spd
.LOR                000000  spd      .RCCGC              010000  spd
.LPLUS              000000  spd      .RCCIA              110000  spd
.LRDLM              000000  spd      .RCENA    000400    000000  spd
.LRPCT              000000  spd      .RCJP               000060  spd
.LSOA               000000  spd      .RCJS               000020  spd
.LSOB               000000  spd      .RCONT              000340  spd
.LSAB               000000  spd      .RCRTN              000240  spd
.LSDO               000000  spd      .RCRY               000400  spd
.LSELE              000000  spd      .RD       001000    000000  spd
.LSKCN              000000  spd      .RJ                 000000  spd
.LXNOR              000000  spd      .RJMAP              000040  spd
.LXOR               000000  spd      .RJZ                000000  spd
.MA                 000017  spd      .RLDCT              000300  spd
.MAND               000007  spd      .RLDLM              230000  spd
.MB                 000017  spd      .RMGC               000[illegible]  spd
.MBAD               000001  spd      .ROENA              000000  spd
.MCCCC              000037  spd      .ROR      030000    000000  spd
.MCCER              000037  spd      .RPLUS              000000  spd
.MCCFZ              000037  spd      .RRDLM              220000  spd
.MCCGC              000037  spd      .RRPCT              000220  spd
.MCCIA              000037  spd      .RSOA     400000    000000  spd
.MCENA              000001  spd      .RSOB     300000    000000  spd
.MCJP               000017  spd      .RSAB     100000    000000  spd
.MCJS               000017  spd      .RSDO     700000    000000  spd
.MCONT              000017  spd      .RSELE              005000  spd
.MCRTN              000017  spd      .RSKCN              240000  spd
.MCRY               000001  spd      .RXNOR    070000    000000  spd
.MD                 000007  spd      .RXOR     060000    000000  spd
.MJ                 007777  spd
.MJMAP              000017  spd
.MJZ                000017  spd
.MLDCT              000017  spd
.MLDLM              000037  spd
.MMGC               001777  spd
.MOENA              000001  spd
.MOR                000007  spd
.MPLUS              000007  spd
.MRDLM              000037  spd
.MRPCT              000017  spd
.MSOA               000007  spd
.MSOB               000007  spd
.MSAB               000007  spd
```

```
                                                                                                                                              SEQ 0650
AEBUF          28#      2702
BBPNT          28#      2774      2950
BEXEC          28#      2696      2924
BUFF           26#      4815      4816      4817      4831      4855      4876      5993      5996      6253      6256      6510      6513
BUFGEN         39#      5914      6172      6436
CALL           176       178       179       278       279       384       386       387       482       483       565       659       660       668
               669       823       824       884       944      1054      1055      1169      1173      1238      1317      1447      1498      1556
              1645      1731      1732      1918      1919      2001      2104      2192      2276      2359      2444      2528      2625      2628
              2774      2950      3019      3111      3235      3366      3442      3521      3592      3670      3675      3789      3794      3799
              4010      4015      4020      4229      4234      4239      4427      4521      4598      4680      4756      4871      5032      5033
              5153      5154      5155      5274      5275      5280      5436      5707      6026      6285      6549      6798      6929      7206
              7352      7603
CEBUF          28#      2700
CRAMPE        3578
CSRCHN        2596      2611
CSRENG         27#
CSRPNT         27#      7530
CSRRQS        2176      2179      2262      2345      2430      2514
DCOMP         3968      3982      3996
E1              67       124       227       328       432       527       595       764       853       909      1117      1469      1588      1955
              2054      3046      3169      4380      4711      4795      4973      5076      5199      5332      5604      5896      6155      6420
              6678      6879      7098      7272
E10            909      1117      1588      1955      2054      2141
E11            527       595       764       909      1117      1469      6678      6879      7098      7272
E12            527       595       764       909      1117      1469      3713      3935      4155      4380      4711
E13            527       595       764       909      1117      1469
E14            527       595       764       909      1117      1469      3320      3397      3473      4380      4550      4711      4795      4973
              5076      5199      5332      5604      5896      6155      6420
E15            527       595       764       909      1117      1469      3397      3473      4973      5076
E16            527       595       764       909      1117      1469      1686      1873      2990      3549
E17            527       595       764       909      1117      1469      4380      4550      4631      4711
E18            983      1686      1873      4465
E19           1686      1873
E2            3046      3169
E20            853      1198      1270      1523      1588      1686      1873      2668
E21            527       595       764
E22             67       124       227       328       432       527       595       764       983      4465
E23           2668      2898
E24           2668      2898
E3            2054      2141      2229      2313      2398      2477      2566
E4            2990      3046      3169      4795      4973      5076      5199      5332      5604      5896      6155      6420
E5              67       124       227       328       432       527
E6             527       595       764      1198      1270      1419      1469      1523      2398      2477      3320      3397      3473      3549
              3623      3713      3935      4155      4380      4631      4711      4973      5076      6678      6879      7098      7272
E7            1955      2054      2229      2313      2398      2477      2668      3046      3169      4795      5332      5604      5896      6155
              6420
E8            2566
E9              67       124       227       328       432       527       595       764       853       983      1588      2054      2141      2229
              2313      2398      2477      2566      3320      3397      3473      3549      3623      3713      3935      4155      4465      6678
              6879      7098      7272
EBUS            65       122       225       326       430       525       593       762       851       907       981      1115      1196      1268
              1417      1467      1521      1586      1684      1871      1953      2052      2139      2227      2311      2396      2475      2564
              2666      2896      2988      3044      3167      3318      3395      3471      3547      3621      3711      3933      4153      4378
```

```
          4463   4548   4629   4709   4793   4971   5074   5197   5330   5602   5894   6153   6418   6676
          6877   7096   7270
EBUSPE    1713   1900   3000   3006
ECOMP     4187   4201   4215
ERESET     32#     76    134    242    352    445    549    616    866    923   1135   1215   1220   1291
          1533   1705   1892   2511   2999   3349   4480   6696   6782   6896   7111   7190   7287   7320
ERFLG       75     78     80    141    158    162    249    261    265    348    366    370    452    465
           469    548    553    607    625    629    642    646    778    788    791    795    799    809
           812    865    872    922    929    932   1002   1011   1019   1034   1041   1132   1141   1144
          1148   1156   1214   1226   1281   1299   1303   1429   1432   1435   1480   1483   1486   1532
          1544   1603   1616   1630   1635   1698   1711   1714   1718   1884   1898   1901   1905   1968
          1977   1989   2072   2082   2091   2158   2168   2180   2245   2255   2261   2263   2328   2338
          2344   2346   2412   2422   2429   2431   2496   2506   2513   2515   2579   2589   2595   2597
          2602   2610   2612   2689   2705   2712   2918   2927   2935   2998   3004   3007   3062   3082
          3089   3184   3206   3213   3335   3344   3351   3353   3412   3421   3427   3429   3491   3500
          3506   3508   3562   3571   3577   3579   3640   3649   3655   3657   3730   3739   3745   3748
          3752   3760   3762   3767   3774   3776   3951   3960   3966   3969   3973   3981   3983   3988
          3995   3997   4170   4179   4185   4188   4192   4200   4202   4207   4214   4216   4397   4406
          4412   4415   4479   4506   4568   4577   4583   4585   4648   4657   4665   4667   4724   4733
          4741   4743   4814   4857   4994   5010   5018   5103   5121   5130   5133   5139   5220   5235
          5244   5247   5253   5259   5262   5352   5362   5369   5623   5633   5640   5916   5926   5933
          6174   6184   6191   6438   6448   6455   6695   6702   6706   6717   6721   6732   6736   6751
          6755   6766   6770   6786   6790   6895   6905   6917   6921   7110   7117   7121   7132   7136
          7148   7152   7164   7168   7180   7184   7194   7198   7286   7296   7308   7312   7328   7340
          7344   7589   7590   7591
GENEPE    1893   3000
IADDR      28#   5412   5683   5984   6244   6501
ICALL     5377   5378   5648   5649   5944   5945   6202   6203   6466   6467
ICALLC    5381   5382   5652   5653   5950   5951   5952   6208   6209   6210   6472   6473   6474
ICSR       28#
IDLE      3747   3761   3775
IEXEC      28#   5359   5630   5923   6181   6445
IEXIT     5385   5656   5948   5955   6206   6213   6470   6477
IFLAG      28#
IIOPF      28#   5410   5681   5977   5994   5998   6235   6240   6254   6257   6491   6497   6511   6515
IIPNT      28#   5436   5707   6026   6285   6549
IJRST     5386   5387   5657   5658   5956   5957   6214   6215   6478   6479
ILAST     5387   5388   5658   5659   5957   5958   6215   6216   6479   6480
INITPI     34#   4819   4996   5104   5221
INTNUM     34#   4849   4879   5017   5127   5129   5241   5243   7630   7636   7638   7641
INTPNO    7636#  7655
INTPNT    5033   5155   5275   7628#
INTPNX    7637   7659#
INTTYP     34#   5131   5245   7646
IPACLR     32#    781    802   1430   1481   1538   1606   2427   2690   2919   3066   3188   3772   3993
          4212   4818   4851   4995   5358   5629   5922   6180   6444   7587
IPASRT     33#   7588
ISETDE    5380   5381   5651   5652   5942   5943   6203   6204   6210   6211   6464   6465
ISETEI    6467   6468   6474   6475
ISETEX    5379   5380   5650   5651   5945   5946   5952   5953   6200   6201   6463   6464
ISETID    5378   5379   5649   5650   5941   5942   6199   6200
ISETIN    5382   5383   5653   5654
ISTART    5383   5384   5654   5655   5946   5947   5953   5954   6204   6205   6211   6212   6468   6469
```

$ARG2

0639

```
          6475   6476
```

0639

```
           6475    6476
IWAIT      5384    5385    5655    5656    5947    5948    5954    5955    6205    6206    6212    6213    6469    6470      SEQ 0652
           6476    6477
LAST         94     179     279     387     483     565     660     669     824     884     945    1055    1169    1173
           1238    1317    1447    1498    1556    1645    1732    1919    2001    2104    2192    2276    2359    2444
           2528    2625    2628    2774    2950    3019    3111    3235    3366    3442    3521    3592    3670    3675
           3789    3794    3799    4010    4015    4020    4229    4234    4239    4427    4521    4598    4680    4756
           4871    5033    5155    5275    5280    5436    5707    6026    6285    6549    6798    6929    7206    7352
           7603
LDCSR        32#    460     618     635     786     807     868     925    1137    1217    1222    1293    1535    1540
           1609    1707    1894    1985    2088    2608    3001    3070    3076    3194    3758    3979    4198    4493
           4502    4663    4739    4830    6007    6013    6266    6272    6524    6530    6713    6728    6746    6762
           6781    6912    7128    7143    7159    7175    7303    7335
LDEBUF       32#    927    1139    1219    1295    1537    1611    1627    1709    1896
LDRAR        32#   3068    3192    4828    6005    6264    6522    6744    6779
M1         4380    4711
M11        1955    2668    2898
M12        2668    2898    3397    3549
M3         3473    3623
MA1          81      91#
MA10        792     813     823#
MA10PN      823     826#
MA11        873     883#
MA12        933     943#
MA13       1038    1054#
MA13PN     1054    1057#
MA14       1145    1167#
MA15       1227    1237#
MA16       1300    1316#
MA17       1436    1446#
MA2         159     175#
MA20       1487    1497#
MA21       1545    1555#
MA23       1715    1731#
MA23PN     1731    1734#
MA24       1902    1918#
MA24PN     1918    1921#
MA2PNA      176     181#
MA2PNB      178     186#
MA3         262     278#
MA34       2599    2624#
MA35       2709    2772#
MA36       2932    2949#
MA37       3008    3018#
MA3PNT      278     281#
MA4         367     385#
MA40       3086    3110#
MA40PN     3111    3113#
MA41       3210    3234#
MA41PN     3235    3237#
MA46       3669#
MA4PNA      384     389#
MA4PNB      386     394#
```

$B

```
MA5         466     482#
```

```
MA5         466     482#
MA54       4510    4520#
MA54PN     4521    4523#
MA5PNT      482     485#
MA6         554     564#
MA60       4861    4871#
MA60PN     4871    4873#
MA64       5366    5435#
MA65       5637    5706#
MA7         626     659#
MA70       5930    6025#
MA71       6188    6284#
MA72       6452    6548#
MA7PNT      659     662#
MB14       1157    1171#
MB22       1632    1644#
MB25       1990    2000#
MB26       2093    2103#
MB27       2181    2191#
MB30       2265    2275#
MB31       2348    2358#
MB32       2433    2443#
MB33       2517    2527#
MB34       2614    2627#
MB42       3355    3365#
MB43       3431    3441#
MB44       3510    3520#
MB45       3581    3591#
MB46       3659    3674#
MB47       3749    3788#
MB50       3970    4009#
MB51       4189    4228#
MB53       4416    4426#
MB55       4587    4597#
MB56       4669    4679#
MB57       4745    4755#
MB61       5021    5031#
MB62       5136    5153#
MB62PN     5153    5157#
MB63       5250    5273#
MB7         643     668#
MB73       6703    6718     6733     6752     6767     6787     6798#
MB73PN     6798    6800#
MB73PP     6801    6805#
MB74       6918    6929#
MB74PN     6929    6931#
MB74PP     6932    6936#
MB75       7118    7133     7149     7165     7181     7195     7206#
MB75PN     7206    7208#
MB75PP     7209    7213#
MB76       7309    7341     7352#
MB76PN     7352    7354#
MB76PP     7355    7359#
```

```
MB7PNT      668     671#
```

0641

$CHR

```
MB7PNT       668       671#                                                                                                                   SEQ 0654
MBCN         135       136       142       145       148       152       155       164       165       170       181       187       243       244
             250       254       257       267       268       273       282       285       291       342       343       349       353       357
             360       364       372       373       378       389       395       446       447       453       458       461       471       472
             477       486       490       495      3072      3199      4836      6009      6268      6526      7518
MBERR       3656
MBUS         851       907      1115      1268      2666      3621
MC47        3764      3778      3793#
MC50        3985      3999      4014#
MC51        4204      4218      4233#
MC63        5263      5279#
MD47        3798#
MD50        4019#
MD51        4238#
MLAST         67       124       227       328       432       527       595       764       853       909       983      1117      1198      1270
            1419      1469      1523      1588      1686      1873      1955      2054      2141      2229      2313      2398      2477      2566
            2668      2898      2990      3046      3169      3320      3397      3473      3549      3623      3713      3935      4155      4380
            4465      4550      4631      4711      4795      4973      5076      5199      5332      5604      5896      6155      6420      6678
            6879      7098      7272
MPNT1        179       279       387       483      7517#
MPNT2       1055      1447      1919      2104      2192      2276      2359      2444      2528      2625      2628      3019      3366      3442
            3521      3592      3675      3789      3794      3799      4010      4015      4020      4229      4234      4239      4427      4598
            4680      4756      5032      5154      5274      5280      6802      6933      7210      7356      7527#     7580
MPNT4        884      1173      1238      1317      1556      1645      7538#
MPNT5        565       660       669       824       945      1169      1498      2001      7557#
MPNT6       1732      3670      7577# [illegible]03
MPRUN       1005      1971      2075 [illegible]61      2248      2331      2415      2499      2582      3069      3193      3338      3415      3494
            3565      3643      3733 [illegible]54      4173      4400      4414      4571      4584      4651      4662      4666      4727      4738
            4742      4829      5004      5114      5229      6006      6265      6523      6899      7290      7321
MSG           91        92        93        94       175       177       383       385       564       883       943       944      1167      1168
            1171      1172      1237      1316      1446      1497      1555      1644      2000      2103      2191      2275      2358      2443
            2527      2624      2627      2772      2773      2949      3018      3110      3234      3365      3441      3520      3591      3669
            3674      3788      3793      3798      4009      4014      4019      4228      4233      4238      4426      4520      4597      4679
            4755      5031      5273      5279      5435      5706      6025      6284      6548      7602
MSTARE      7592      7602#
MSTART      1008      1613      1974      2079      2165      2252      2335      2419      2503      2586      3341      3418      3497      3568
            3646      3736      3957      4176      4403      4574      4654      4730      5007      5118      5232      6902      7293      7325
            7587#
NDMP         981      1586      1953      2052      2139      2227      2311      2396      2475      2564      2666      2896      3044      3167
            3318      3395      3471      3547      3621      3711      3933      4153      4378      4548      4629      4709      4793      4971
            5074      5197      5330      5602      5894      6153      6418      6676      6877      7270
ODELAY        27#     4843      5016
P           5414      5685      5970      5988      6228      6248      6505      7594      7596      7597
PAT         5913      6171      6435
PCLEAR      1434      4662
PF500         35#     7614
PF501         35#     7617
PF502         35#     7620
PFAIL         35#     1029      1033      4481      4487      4496      4505
PFAPR         35#     7611
PFPNT       1066      4524      7610#
PISYOF        35#     4822
PISYON        35#     4821      4847
```

```
RDCSR         32#       77       146       153       255       354       361       619       636       787       808       928      1140      1431
RDCSR       32#    77    146    153    255    354    361    619    636    787    808    928   1140   1431
          1482   1710   1897   2089   2174   2260   2343   2428   2512   2594   2609   3003   3350   3426
          3505   3576   3654   3744   3759   3773   3965   3980   3994   4184   4199   4213   4411   4485
          4494   4503   4582   4664   4740   5019   5134   5248   5258   6699   6714   6729   6747   6763
          6783   7114   7129   7144   7160   7176   7191   7577
RDEBUF      32#   869   1153   1223   1296   1541   1628   1986   3002   3077   6913   7304   7336
RQINT     3352   3428   3507   5261
SAVCS1     617    663    672    785    806    827   7671#
SAVCS2     546    552    623    640    783    790    804    811    920    931   1127   1143   1478   1485
          1983   1988   7562   7672#
SAVCS3     551    621    638    789    810    930   1142   1484   1987   7568   7673#
SAVCS4    7674#
SAVCSR    1433   1712   1899   2092   2178   2264   2347   2432   2516   2598   2607   2613   3005   3354
          3430   3509   3580   3658   3746   3757   3763   3777   3967   3978   3984   3998   4186   4197
          4203   4217   4413   4586   4668   4744   5260   7527   7579   7670#
SAVDA1     149    157    193    253    260    294    356    365    398    457    464    498    870   1025
          1058   3204   3242   7667#
SAVDA2     151    190    259    297    363    401    463    501   1032   1062   3202   3248   7668#
SAVDA3     156    196   7669#
SAVDAT    1704   1708   1735   1891   1895   1922   5098   5099   5110   5115   5132   5143   5158   7666#
SAVEB1    1128   1155   1212   1225   1290   1294   1298   1530   1620   3080   3118   7543   7675#
SAVEB2    1154   1224   1297   1542   1631   3078   3124   7549   7676#
SAVEBU    6915#  6915   6931   7306   7338   7354
SAVMBC     135#   135    142    145    148    152    155    170    243    250    254    257    273    342
           349    353    357    360    364    378    446    458    461    477
SCOSW       39#  3113   3237   7538   7557
SDATA       33#  1006   1608   1972   2076   2162   2249   2332   2416   2500   2583   3339   3416   3495
          3566   3644   3734   3955   4174   4401   4572   4652   4728   5005   5116   5230   5986   6246
          6503   6900   7291   7322
SETVEC      34#  4820   4997   5105   5222
SINCYC    4738
SNEXT       33#  1007   1612   1973   2078   2164   2251   2334   2418   2502   2585   3340   3417   3496
          3567   3645   3735   3956   4175   4402   4573   4653   4729   5006   5117   5231   6901   7292
          7324
SSLAST    2768   2769   2945   2946
SSSTRT    2720   2721   2722   2723   2724   2725   2726   2727   2728   2729   2730   2731   2732   2733
          2734   2735   2736   2737   2738   2739   2740   2741   2742   2743   2744   2745   2746   2747
          2748   2749   2750   2751   2752   2753   2754   2755   2756   2757   2758   2759   2760   2761
          2762   2763   2764   2765   2766   2767   2768   2943   2944   2945
T13M       982    996   1071#
T16PAT    1284   1321#
T22M      1587   1597   1649#
T23PAT    1022   1701   1742#  1887
T25M      1954   1962   2005#
T26M      2053   2066   2108#
T27M      2140   2152   2196#
T30M      2228   2239   2280#
T31M      2312   2322   2363#
T32M      2397   2406   2448#
T33M      2476   2490   2532#
T34M      2565   2573   2632#
T35M      2667   2683   2778#
T36M      2897   2912   2954#
```

```
T40DAT    3060   3064   3094   3103#
```

```
T40DAT   3060   3064   3094   3103#                                          SEQ 0656
T40M     3045   3054   3130#
T41DAT   3182   3186   3190   3218   3227#
T41M     3168   3176   3254#
T42M     3319   3329   3370#
T43M     3396   3406   3446#
T44M     3472   3485   3526#
T45M     3548   3556   3596#
T46M     3622   3634   3680#
T47M     3712   3724   3804#
T50M     3934   3945   4025#
T51M     4154   4164   4244#
T53M     4379   4391   4431#
T55M     4549   4562   4602#
T56M     4630   4642   4684#
T57M     4710   4718   4760#
T60M     4794   4808   4885#
T61M     4972   4988   5037#
T62M     5075   5092   5164#
T63M     5198   5214   5284#
T64ADD   5381   5397#
T64ADI   5377   5394#
T64ADR   5394#  5394   5397
T64ADT   5411   5419#
T64M     5331   5346   5447#
T65ADD   5652   5668#
T65ADI   5648   5665#
T65ADR   5665#  5665   5668
T65ADT   5682   5690#
T65M     5603   5617   5718#
T70ADD   5951   5973#
T70ADI   5950   5968#
T70ADR   5968#  5968   5973
T70ADT   5983   6018#
T70M     5895   5905   6032#
T71ADD   6209   6231#
T71ADI   6208   6226#
T71ADR   6226#  6226   6231   6242
T71ADT   6243   6277#
T71M     6154   6163   6291#
T72ADD   6473   6487#
T72ADI   6472   6484#
T72ADR   6484#  6484   6487   6499
T72ADT   6500   6535#
T72M     6419   6427   6555#
T73M     6677   6688   6826#
T74M     6878   6888   6955#
T76M     7271   7279   7381#
T76SAV   7317#  7317   7318
TA1        75#    82
TA10      778#   793
TA11      865#   874
TA12      922#   934
```

```
TA13        1002#    1009
TA14        1132#    1146     1158
TA15        1214#    1228
TA16        1281#    1301     1308
TA17        1429#    1437
TA2          140#     166
TA20        1480#    1488
TA21        1532#    1546
TA22        1603#    1614     1633
TA23        1698#    1716     1723
TA24        1884#    1903     1910
TA25        1968#    1975     1991
TA26        2072#    2080     2094
TA27        2158#    2166     2182
TA2L         141#     160
TA2X         143      164#
TA3          248#     269
TA30        2245#    2253     2256
TA31        2328#    2336     2349
TA32        2412#    2420     2434
TA33        2496#    2504     2518
TA34        2579#    2587     2600     2615
TA35        2696#    2698     2712
TA36        2924#    2926     2935
TA37        2998#    3009
TA3L         249#     263
TA3X         251      267#
TA4          347#     374
TA40        3062#    3087     3095
TA41        3184#    3211     3219
TA42        3335#    3342     3356
TA43        3412#    3419     3432
TA44        3491#    3498     3511
TA45        3562#    3569     3582
TA46        3640#    3647     3660
TA47        3730#    3737     3750     3765     3779
TA4L         348#     368
TA4X         350      372#
TA5          451#     473
TA50        3951#    3958     3971     3986     4000
TA51        4170#    4177     4190     4205     4219
TA53        4397#    4404     4417
TA54        4479#    4511
TA55        4568#    4575     4588
TA56        4648#    4655     4670
TA57        4724#    4731     4746
TA5L         452#     467
TA5X         454      471#
TA6          548#     555
TA61        4994#    5008     5022
TA62        5103#    5119     5137     5145
TA63        5220#    5233     5251     5264
TA64        5358#    5369
```

```
TA65        5629#   5640
TA7          607#    627     644     651
TA70        5922#   5933
TA71        6180#   6191
TA72        6444#   6455
TA73        6695#   6704    6719    6734    6753    6768    6788
TA74        6895#   6903    6919
TA75        7110#   7119    7134    7150    7166    7182    7196
TA76        7286#   7294    7310
TA7PAT       613     622     639     677#
TB10         799#    814
TB13        1019#   1039    1046
TB22        1622#   1636
TB27        2174#   2177
TB42        3348#
TB43        3425#
TB44        3504#
TB45        3575#
TB46        3653#
TB47        3743#
TB50        3964#
TB51        4183#
TB53        4410#
TB54        4488    4497    4506#
TB55        4581#
TB56        4661#
TB57        4737#
TB61        5014#
TB62        5125#
TB62A       5127#   5128
TB63        5241#   5242
TB64        5359#   5361
TB65        5630#   5632
TB70        5923#   5925
TB71        6181#   6183
TB72        6445#   6447
TB76        7318#   7326    7342
TC47        3756#
TC50        3977#
TC51        4196#
TC63        5257#
TD47        3771#
TD50        3992#
TD51        4211#
TG1           64      72#
TG10         761     773#
TG11         850     862#
TG12         906     917#
TG13         980     988#
TG14        1114    1124#
TG15        1195    1204#
TG16        1267    1275#
TG17        1416    1426#
```

```
TG2       121     132#
TG20     1466    1475#
TG21     1520    1528#
TG22     1585    1595#
TG23     1683    1692#
TG24     1870    1878#
TG25     1952    1960#
TG26     2051    2064#
TG27     2138    2150#
TG3       224     234#
TG30     2226    2237#
TG31     2310    2320#
TG32     2395    2404#
TG33     2474    2482#
TG34     2563    2571#
TG35     2665    2681#
TG36     2895    2910#
TG37     2987    2995#
TG4       325     334#
TG40     3043    3052#
TG41     3166    3174#
TG42     3317    3327#
TG43     3394    3404#
TG44     3470    3483#
TG45     3546    3554#
TG46     3620    3632#
TG47     3710    3722#
TG5       429     437#
TG50     3932    3943#
TG51     4152    4162#
TG53     4377    4389#
TG54     4462    4470#
TG55     4547    4560#
TG56     4628    :640#
TG57     4708    4716#
TG6       524     537#
TG60     4792    4800#
TG61     4970    4980#
TG62     5073    5084#
TG63     5196    5206#
TG64     5329    5338#
TG65     5601    5609#
TG7       592     601#
TG70     5893    5903#
TG71     6152    6161#
TG72     6417    6425#
TG73     6675    6686#
TG74     6876    6886#
TG75     7095    7104#
TG76     7269    7277#
TL35     2689#   2710
TL36     2918#   2933
TL60     4814#   4862
```

```
TL64       5352#   5367
TL65       5623#   5638
TL70       5916#   5931
TL71       6174#   6189
TL72       6438#   6453
TLOAD        27#    997    1598    1963    2067    2153    2240    2323    2407    2491    2574    2684    2913    3055
           3177    3330    3407    3486    3557    3635    3725    3946    4165    4392    4563    4643    4719    4809
           4989    5093    5215    5347    5618    5906    6164    6428    6689    6889    7280
TRACE        27#     73     133     241     341     444     544     602     774     863     918     995    1125    1211
           1276    1427    1476    1529    1596    1693    1879    1961    2065    2151    2238    2321    2405    2489
           2572    2682    2911    2996    3053    3175    3328    3405    3484    3555    3633    3723    3944    4163
           4390    4477    4561    4641    4717    4807    4987    5091    5213    5345    5616    5904    6162    6426
           6687    6887    7105    7278
TS35       2691    2720#
TS36       2920    2943#
TS64       5353    5377#
TS64A      5381#   5386
TS64Z      5378    5379    5380    5389#
TS65       5624    5648#
TS65A      5652#   5657
TS65Z      5649    5650    5651    5660#
TS70       5917    5941#
TS70A      5951#   5956
TS70AA     5945    5952    5963#   5982    5997    5999
TS70EX     5944    5993#
TS70Z      5941    5942    5960#
TS71       6175    6199#
TS71A      6209#   6214
TS71AA     6203    6210    6221#   6241    6258
TS71DE     6202    6253#
TS71Z      6199    6200    6218#
TS72       6439    6463#
TS72A      6473#   6478
TS72AA     6467    6474    6496    6498    6514    6516    6543#
TS72EX     6466    6510#
TS72Z      6463    6464    6540#
TSTA1        21#   2670    2899
TSTA2        21#   2671    2900
TSTA3        21#   2672    2901
TSTA4        21#   2673    2902
TSTA5        21#   2674    2903
TSTA6        21#   2675    2904
TSTA7        21#   2676    2905
TSTE1        10      64#
TSTE10       10     529     596     761#
TSTE11       11     530     765     850#
TSTE12       11     531     766     854     906#
TSTE13       11     980#
TSTE14       11     532     767     855     910    1114#
TSTE15       11     768     856     911    1118    1195#
TSTE16       11     857     912    1119    1199    1267#
TSTE17       11    1416#
TSTE2        10     121#
```

```
TSTE20      11      1420    1466#
TSTE21      12      1421    1470    1520#
TSTE22      12      1585#
TSTE23      12      1683#
TSTE24      12      1687    1870#
TSTE25      12      1952#
TSTE26      12      2051#
TSTE27      12      2055    2138#
TSTE3       10       125     224#
TSTE30      12      2056    2142    2226#
TSTE31      13      2057    2143    2230    2310#
TSTE32      13      2058    2144    2231    2314    2395#
TSTE33      13      2059    2145    2232    2315    2399    2474#
TSTE34      13      2563#
TSTE35      13      2665#
TSTE36      13      2669    2895#
TSTE37      13      2987#
TSTE4       10       126     228     325#
TSTE40      13      3043#
TSTE41      14      3047    3166#
TSTE42      14      3317#
TSTE43      14      3321    3394#
TSTE44      14      3322    3398    3470#
TSTE45      14      3399    3546#   4384
TSTE46      14      3474    3620#
TSTE47      14      3710#
TSTE5       10       127     229     329     429#
TSTE50      14      3714    3932#
TSTE51      15      3715    3936    4152#
TSTE52      15      4352#
TSTE53      15      4377#
TSTE54      15      4462#
TSTE55      15      1589    4547#
TSTE56      15      1590    4551    4628#
TSTE57      15      4708#
TSTE6       10       524#
TSTE60      15      4792#
TSTE61      16      4970#
TSTE62      16      4974    5073#
TSTE63      16      4975    5077    5196#
TSTE64      16      5078    5200    5329#
TSTE65      16      5079    5201    5333    5601#
TSTE66      16      5841#
TSTE67      16      5854#
TSTE7       10       528     592#
TSTE70      16      5893#
TSTE71      17      5897    6152#
TSTE72      17      5898    6156    6417#
TSTE73      17      6675#
TSTE74      17      3716    3937    4156    6679    6876#
TSTE75      17      6680    6880    7095#
TSTE76      17      3717    3938    4157    6681    6881    7099    7269#
TSTEBF     867      1136    1216    1221    1292    1534    1539    1607    1706    1893    1984    3000    3075    6911
```

```
           7302    7334
TSTFLG       26#    235     238     335     338     438     441     538     541     989     992    1205    1208    2483      SEQ 0662
           2486    4471    4474    4801    4804    4981    4984    5085    5088    5207    5210    5339    5342    5610
           5613
TSTPC        26#
TSTSUB       26#    140     248     347     451     612     633     780     801    1004    1021    1134    1152    1283
           1605    1622    1700    1886    1970    1981    2074    2086    2160    2172    2247    2259    2330    2342
           2414    2426    2498    2510    2581    2593    2606    2692    2931    3065    3187    3337    3348    3414
           3425    3493    3504    3564    3575    3642    3653    3732    3743    3756    3771    3953    3964    3977
           3992    4172    4183    4196    4211    4399    4410    4570    4581    4650    4661    4726    4737    5003
           5014    5113    5126    5228    5240    5257    5354    5625    5918    6176    6440    6698    6711    6726
           6742    6760    6777    6800    6898    6910    7113    7126    7141    7157    7173    7189    7208    7289
           7301    7316    7319    7333
TSTU10       22#   3475    3624    4552    4632
TSTU11       22#   3476    3625    4553    4633
TSTU12       22#   3477    3626    4554    4634
TSTU13       22#   3478    3627    4555    4635
TSTU41       22#   4383
TSTU6        22#   4381
TSTU7        22#   4382
TUSER       236     336     439     539     990    1206    2484    4472    4802    4982    5086    5208    5340    5611
TX1          83      87#
TX10        794     795     815     819#
TX11        875     879#
TX12        935     939#
TX13       1010    1011    1024    1040    1041    1050#
TX14       1147    1148    1159    1163#
TX15       1229    1233#
TX16       1286    1302    1303    1312#
TX17       1438    1442#
TX2         161     162     170#
TX20       1489    1493#
TX21       1547    1551#
TX22       1615    1616    1634    1635    1640#
TX23       1703    1717    1718    1727#
TX24       1889    1904    1905    1914#
TX25       1976    1977    1992    1996#
TX26       2081    2082    2095    2099#
TX27       2167    2168    2183    2187#
TX3         264     265     273#
TX30       2254    2255    2267    2271#
TX31       2337    2338    2350    2354#
TX32       2421    2422    2435    2439#
TX33       2505    2506    2519    2523#
TX34       2588    2589    2601    2602    2616    2620#
TX35       2697    2711    2716#
TX36       2925    2934    2939#
TX37       3010    3014#
TX4         369     370     378#
TX40       3088    3089    3099#
TX41       3212    3213    3223#
TX42       3343    3344    3357    3361#
TX43       3420    3421    3433    3437#
```

```
TX44          3499       3500       3512       3516#
TX45          3570       3571       3583       3587#
TX46          3648       3649       3661       3665#
TX47          3738       3739       3751       3752       3766       3767       3780       3784
TX5            468        469        477#
TX50          3959       3960       3972       3973       3987       3988       4001       4005#
TX51          4178       4179       4191       4192       4206       4207       4220       4224#
TX53          4405       4406       4418       4422#
TX54          4512       4516#
TX55          4576       4577       4589       4593#
TX56          4656       4657       4671       4675#
TX57          4732       4733       4747       4751#
TX6            556        560#
TX60          4863       4867#
TX61          5009       5010       5023       5027#
TX62          5120       5121       5138       5139       5149#
TX63          5234       5235       5252       5253       5265       5269#
TX64          5360       5368       5373#
TX65          5631       5639       5644#
TX7            614        628        629        645        646        655#
TX70          5924       5932       5937#
TX71          6182       6190       6195#
TX72          6446       6454       6459#
TX73          6705       6706       6720       6721       6735       6736       6754       6755       6769       6770       6789       6790       6794#
TX74          6904       6905       6920       6921       6925#
TX75          7120       7121       7135       7136       7151       7152       7167       7168       7183       7184       7197       7198       7202#
TX76          7295       7296       7311       7312       7327       7328       7343       7344       7348#
TXALL          565        660        669        824        884        945       1055       1169       1173       1238       1317       1447       1498       1556
              1645       1732       1919       2001       2104       2192       2276       2359       2444       2528       2625       2628       2774       2950
              3019       3111       3235       3366       3442       3521       3592       3670       3675       3789       3794       3799       4010       4015
              4020       4229       4234       4239       4427       4521       4598       4680       4756       4871       5032       5033       5154       5155
              5274       5275       5280       5436       5707       6026       6285       6549       6798       6929       7206       7352       7602       7603
TXNOT           91         92         93        175        176        177        178        278        383        384        385        386        482        564
               659        668        823        883        943        944       1054       1167       1168       1171       1172       1237       1316       1446
              1497       1555       1644       1731       1918       2000       2103       2191       2275       2358       2443       2527       2624       2627
              2772       2773       2949       3018       3110       3234       3365       3441       3520       3591       3669       3674       3788       3793
              3798       4009       4014       4019       4228       4233       4238       4426       4520       4597       4679       4755       5031       5153
              5273       5279       5435       5706       6025       6284       6548
TXTINH        3114       3116       3120       3122       3238       3240       3244       3246       7539       7541       7545       7547       7558       7560
              7564       7566
TXYES           94        179        279        387        483
UDEBUG          26#       237        337        440        540        991       1207       2485       3073       3200       4473       4803       4832       4837
              4983       5087       5209       5341       5612       6000       6010       6259       6269       6517       6527
USER           234        334        437        537        988       1204       2482       4470       4800       4980       5084       5206       5338       5609
              5969       6227
Z2              45#        72        132        240        340        443        543        601        773        862        917        994       1124       1210
              1275       1426       1475       1528       1595       1692       1878       1960       2064       2150       2237       2320       2404       2488
              2571       2681       2910       2995       3052       3174       3327       3404       3483       3554       3632       3722       3943       4162
              4389       4476       4560       4640       4716       4806       4986       5090       5212       5344       5615       5903       6161       6425
              6686       6886       7104       7277
ZEBUS           65        122        225        326        430        525        593        762        851        907        981       1115       1196       1268
              1417       1467       1521       1586       1684       1871       1953       2052       2139       2227       2311       2396       2475       2564
              2666       2896       2988       3044       3167       3318       3395       3471       3547       3621       3711       3933       4153       4353
```

```
651
         4378  4463  4548  4629  4709  4793  4971  5074  5197  5330  5602  5842  5855  5894      SEQ 0664
         6153  6418  6676  6877  7096  7270
$ARG2    1071# 1071  1073# 1073  1075# 1075  1077# 1077  1079# 1079  1081# 1081  1083# 1083
         1085# 1085  1087# 1087  1649# 1649  2005# 2005  2007# 2007  2009# 2009  2011# 2011
         2013# 2013  2018# 2018  2020# 2020  2022# 2022  2024# 2024  2026# 2026  2108# 2108
         2110# 2110  2112# 2112  2196# 2196  2198# 2198  2200# 2200  2202# 2202  2280# 2280
         2282# 2282  2284# 2284  2286# 2286  2363# 2363  2365# 2365  2367# 2367  2369# 2369
         2371# 2371  2373# 2373  2448# 2448  2450# 2450  2452# 2452  2532# 2532  2534# 2534
         2536# 2536  2632# 2632  2634# 2634  2636# 2636  2638# 2638  2778# 2778  2780# 2780
         2782# 2782  2784# 2784  2786# 2786  2788# 2788  2790# 2790  2792# 2792  2794# 2794
         2796# 2796  2798# 2798  2800# 2800  2802# 2802  2804# 2804  2806# 2806  2808# 2808
         2810# 2810  2812# 2812  2814# 2814  2816# 2816  2818# 2818  2820# 2820  2822# 2822
         2824# 2824  2826# 2826  2828# 2828  2830# 2830  2832# 2832  2834# 2834  2836# 2836
         2838# 2838  2840# 2840  2842# 2842  2844# 2844  2846# 2846  2848# 2848  2850# 2850
         2852# 2852  2854# 2854  2856# 2856  2858# 2858  2860# 2860  2862# 2862  2864# 2864
         2866# 2866  2868# 2868  2870# 2870  2872# 2872  2954# 2954  2956# 2956  2958# 2958
         2960# 2960  3130# 3130  3132# 3132  3134# 3134  3136# 3136  3138# 3138  3140# 3140
         3254# 3254  3256# 3256  3258# 3258  3260# 3260  3263# 3263  3266# 3266  3268# 3268
         3270# 3270  3272# 3272  3275# 3275  3277# 3277  3279# 3279  3281# 3281  3283# 3283
         3288# 3288  3290# 3290  3292# 3292  3294# 3294  3296# 3296  3370# 3370  3372# 3372
         3374# 3374  3446# 3446  3448# 3448  3450# 3450  3526# 3526  3596# 3596  3598# 3598
         3600# 3600  3680# 3680  3804# 3804  3806# 3806  3808# 3808  3810# 3810  3815# 3815
         3817# 3817  3819# 3819  3824# 3824  3826# 3826  3831# 3831  3833# 3833  3835# 3835
         3837# 3837  3842# 3842  3844# 3844  3846# 3846  3851# 3851  3857# 3857  3859# 3859
         3864# 3864  3866# 3866  3868# 3868  3870# 3870  3875# 3875  3877# 3877  3879# 3879
         3884# 3884  3886# 3886  3891# 3891  3898# 3898  3900# 3900  3902# 3902  4025# 4025
         4027# 4027  4029# 4029  4031# 4031  4036# 4036  4038# 4038  4040# 4040  4045# 4045
         4047# 4047  4052# 4052  4054# 4054  4056# 4056  4058# 4058  4063# 4063  4065# 4065
         4067# 4067  4072# 4072  4078# 4078  4080# 4080  4085# 4085  4087# 4087  4089# 4089
         4091# 4091  4096# 4096  4098# 4098  4100# 4100  4105# 4105  4107# 4107  4112# 4112
         4119# 4119  4121# 4121  4123# 4123  4244# 4244  4246# 4246  4248# 4248  4250# 4250
         4255# 4255  4257# 4257  4259# 4259  4264# 4264  4266# 4266  4271# 4271  4273# 4273
         4275# 4275  4277# 4277  4282# 4282  4284# 4284  4286# 4286  4291# 4291  4297# 4297
         4299# 4299  4304# 4304  4306# 4306  4308# 4308  4310# 4310  4315# 4315  4317# 4317
         4319# 4319  4324# 4324  4326# 4326  4331# 4331  4338# 4338  4340# 4340  4342# 4342
         4431# 4431  4602# 4602  4604# 4604  4606# 4606  4684# 4684  4760# 4760  4885# 4885
         4890# 4890  4892# 4892  4894# 4894  4896# 4896  4898# 4898  4900# 4900  4902# 4902
         4904# 4904  4906# 4906  4908# 4908  4910# 4910  4915# 4915  4917# 4917  4919# 4919
         4921# 4921  4926# 4926  4928# 4928  4930# 4930  4932# 4932  4934# 4934  4936# 4936
         4938# 4938  4940# 4940  4942# 4942  4944# 4944  5037# 5037  5039# 5039  5041# 5041
         5164# 5164  5166# 5166  5168# 5168  5284# 5284  5286# 5286  5288# 5288  5447# 5447
         5449# 5449  5451# 5451  5453# 5453  5455# 5455  5457# 5457  5462# 5462  5464# 5464
         5466# 5466  5468# 5468  5473# 5473  5475# 5475  5477# 5477  5479# 5479  5484# 5484
         5486# 5486  5488# 5488  5490# 5490  5495# 5495  5497# 5497  5499# 5499  5501# 5501
         5506# 5506  5508# 5508  5510# 5510  5512# 5512  5517# 5517  5519# 5519  5521# 5521
         5523# 5523  5531# 5531  5533# 5533  5535# 5535  5537# 5537  5539# 5539  5541# 5541
         5547# 5547  5549# 5549  5551# 5551  5553# 5553  5558# 5558  5560# 5560  5718# 5718
         5720# 5720  5722# 5722  5724# 5724  5726# 5726  5728# 5728  5733# 5733  5735# 5735
         5737# 5737  5739# 5739  5744# 5744  5746# 5746  5748# 5748  5750# 5750  5755# 5755
         5757# 5757  5759# 5759  5761# 5761  5766# 5766  5768# 5768  5770# 5770  5772# 5772
         5777# 5777  5779# 5779  5781# 5781  5783# 5783  5788# 5788  5790# 5790  5792# 5792
         5794# 5794  5802# 5802  5804# 5804  5806# 5806  5808# 5808  5810# 5810  5812# 5812
         5818# 5818  5820# 5820  5822# 5822  5824# 5824  5829# 5829  5831# 5831  6032# 6032
```

```
.LJZ
.LLDCT

.LLDLM
.LMGC

.LOENA

.LOR

.LPLUS

.LRDLM
.LRPCT

.LSOA
```

```
652      6034# 6034  6036# 6036  6041# 6041  6043# 6043  6045# 6045  6050# 6050  6052# 6052
```

```
            6034#  6034   6036#  6036   6041#  6041   6043#  6043   6045#  6045   6050#  6050   6052#  6052      SEQ 0665
            6054#  6054   6059#  6059   6061#  6061   6066#  6066   6068#  6068   6070#  6070   6072#  6072
            6074#  6074   6080#  6080   6082#  6082   6084#  6084   6091#  6091   6093#  6093   6095#  6095
            6097#  6097   6099#  6099   6101#  6101   6103#  6103   6105#  6105   6107#  6107   6112#  6112
            6114#  6114   6291#  6291   6293#  6293   6295#  6295   6300#  6300   6302#  6302   6304#  6304
            6309#  6309   6311#  6311   6313#  6313   6318#  6318   6320#  6320   6325#  6325   6327#  6327
            6329#  6329   6331#  6331   6333#  6333   6339#  6339   6341#  6341   6343#  6343   6351#  6351
            6353#  6353   6355#  6355   6357#  6357   6359#  6359   6361#  6361   6363#  6363   6365#  6365
            6367#  6367   6369#  6369   6371#  6371   6373#  6373   6378#  6378   6380#  6380   6555#  6555
            6557#  6557   6559#  6559   6564#  6564   6566#  6566   6568#  6568   6573#  6573   6575#  6575
            6577#  6577   6582#  6582   6584#  6584   6589#  6589   6591#  6591   6593#  6593   6595#  6595
            6597#  6597   6603#  6603   6605#  6605   6607#  6607   6614#  6614   6616#  6616   6618#  6618
            6620#  6620   6622#  6622   6624#  6624   6626#  6626   6628#  6628   6630#  6630   6635#  6635
            6637#  6637   6826#  6826   6828#  6828   6830#  6830   6832#  6832   6834#  6834   6836#  6836
            6838#  6838   6840#  6840   6955#  6955   6957#  6957   6959#  6959   6961#  6961   6963#  6963
            6968#  6968   6970#  6970   6972#  6972   6974#  6974   6976#  6976   6978#  6978   6984#  6984
            6986#  6986   6988#  6988   6990#  6990   6993#  6993   6995#  6995   6997#  6997   6999#  6999
            7001#  7001   7003#  7003   7009#  7009   7011#  7011   7013#  7013   7015#  7015   7018#  7018
            7020#  7020   7022#  7022   7024#  7024   7026#  7026   7028#  7028   7034#  7034   7036#  7036
            7038#  7038   7040#  7040   7043#  7043   7045#  7045   7047#  7047   7049#  7049   7051#  7051
            7053#  7053   7055#  7055   7060#  7060   7062#  7062   7381#  7381   7383#  7383   7385#  7385
            7387#  7387   7389#  7389   7394#  7394   7396#  7396   7398#  7398   7400#  7400   7402#  7402
            7404#  7404   7410#  7410   7412#  7412   7414#  7414   7416#  7416   7419#  7419   7421#  7421
            7423#  7423   7425#  7425   7427#  7427   7429#  7429   7435#  7435   7437#  7437   7439#  7439
            7441#  7441   7444#  7444   7446#  7446   7448#  7448   7450#  7450   7452#  7452   7454#  7454
            7456#  7456   7461#  7461   7463#  7463   7465#  7465   7467#  7467   7469#  7469   7471#  7471
            7473#  7473   7479#  7479   7481#  7481   7483#  7483   7485#  7485   7488#  7488   7490#  7490
            7492#  7492   7494#  7494   7496#  7496   7498#  7498   7500#  7500   7505#  7505   7507#  7507
$B          1071#  1071   1073#  1073   1075#  1075   1077#  1077   1079#  1079   1081#  1081   1083#  1083
            1085#  1085   1087#  1087   1649#  1649   2005#  2005   2007#  2007   2009#  2009   2011#  2011
            2013#  2013   2018#  2018   2020#  2020   2022#  2022   2024#  2024   2026#  2026   2108#  2108
            2110#  2110   2112#  2112   2196#  2196   2198#  2198   2200#  2200   2202#  2202   2280#  2280
            2282#  2282   2284#  2284   2286#  2286   2363#  2363   2365#  2365   2367#  2367   2369#  2369
            2371#  2371   2373#  2373   2448#  2448   2450#  2450   2452#  2452   2532#  2532   2534#  2534
            2536#  2536   2632#  2632   2634#  2634   2636#  2636   2638#  2638   2778#  2778   2780#  2780
            2782#  2782   2784#  2784   2786#  2786   2788#  2788   2790#  2790   2792#  2792   2794#  2794
            2796#  2796   2798#  2798   2800#  2800   2802#  2802   2804#  2804   2806#  2806   2808#  2808
            2810#  2810   2812#  2812   2814#  2814   2816#  2816   2818#  2818   2820#  2820   2822#  2822
            2824#  2824   2826#  2826   2828#  2828   2830#  2830   2832#  2832   2834#  2834   2836#  2836
            2838#  2838   2840#  2840   2842#  2842   2844#  2844   2846#  2846   2848#  2848   2850#  2850
            2852#  2852   2854#  2854   2856#  2856   2858#  2858   2860#  2860   2862#  2862   2864#  2864
            2866#  2866   2868#  2868   2870#  2870   2872#  2872   2954#  2954   2956#  2956   2958#  2958
            2960#  2960   3130#  3130   3132#  3132   3134#  3134   3136#  3136   3138#  3138   3140#  3140
            3254#  3254   3256#  3256   3258#  3258   3260#  3260   3263#  3263   3266#  3266   3268#  3268
            3270#  3270   3272#  3272   3275#  3275   3277#  3277   3279#  3279   3281#  3281   3283#  3283
            3288#  3288   3290#  3290   3292#  3292   3294#  3294   3296#  3296   3370#  3370   3372#  3372
            3374#  3374   3446#  3446   3448#  3448   3450#  3450   3526#  3526   3596#  3596   3598#  3598
            3600#  3600   3680#  3680   3804#  3804   3806#  3806   3808#  3808   3810#  3810   3815#  3815
            3817#  3817   3819#  3819   3824#  3824   3826#  3826   3831#  3831   3833#  3833   3835#  3835
            3837#  3837   3842#  3842   3844#  3844   3846#  3846   3851#  3851   3857#  3857   3859#  3859
            3864#  3864   3866#  3866   3868#  3868   3870#  3870   3875#  3875   3877#  3877   3879#  3879
            3884#  3884   3886#  3886   3891#  3891   3898#  3898   3900#  3900   3902#  3902   4025#  4025
            4027#  4027   4029#  4029   4031#  4031   4036#  4036   4038#  4038   4040#  4040   4045#  4045
```

```
            4047#  4047   4052#  4052   4054#  4054   4056#  4056   4058#  4058   4063#  4063   4065#  4065
```

653

```
4047#  4047    4052#  4052    4054#  4054    4056#  4056    4058#  4058    4063#  4063    4065#  4065
4067#  4067    4072#  4072    4078#  4078    4080#  4080    4085#  4085    4087#  4087    4089#  4089
4091#  4091    4096#  4096    4098#  4098    4100#  4100    4105#  4105    4107#  4107    4112#  4112
4119#  4119    4121#  4121    4123#  4123    4244#  4244    4246#  4246    4248#  4248    4250#  4250
4255#  4255    4257#  4257    4259#  4259    4264#  4264    4266#  4266    4271#  4271    4273#  4273
4275#  4275    4277#  4277    4282#  4282    4284#  4284    4286#  4286    4291#  4291    4297#  4297
4299#  4299    4304#  4304    4306#  4306    4308#  4308    4310#  4310    4315#  4315    4317#  4317
4319#  4319    4324#  4324    4326#  4326    4331#  4331    4338#  4338    4340#  4340    4342#  4342
4431#  4431    4602#  4602    4604#  4604    4606#  4606    4684#  4684    4760#  4760    4885#  4885
4890#  4890    4892#  4892    4894#  4894    4896#  4896    4898#  4898    4900#  4900    4902#  4902
4904#  4904    4906#  4906    4908#  4908    4910#  4910    4915#  4915    4917#  4917    4919#  4919
4921#  4921    4926#  4926    4928#  4928    4930#  4930    4932#  4932    4934#  4934    4936#  4936
4938#  4938    4940#  4940    4942#  4942    4944#  4944    5037#  5037    5039#  5039    5041#  5041
5164#  5164    5166#  5166    5168#  5168    5284#  5284    5286#  5286    5288#  5288    5447#  5447
5449#  5449    5451#  5451    5453#  5453    5455#  5455    5457#  5457    5462#  5462    5464#  5464
5466#  5466    5468#  5468    5473#  5473    5475#  5475    5477#  5477    5479#  5479    5484#  5484
5486#  5486    5488#  5488    5490#  5490    5495#  5495    5497#  5497    5499#  5499    5501#  5501
5506#  5506    5508#  5508    5510#  5510    5512#  5512    5517#  5517    5519#  5519    5521#  5521
5523#  5523    5531#  5531    5533#  5533    5535#  5535    5537#  5537    5539#  5539    5541#  5541
5547#  5547    5549#  5549    5551#  5551    5553#  5553    5558#  5558    5560#  5560    5718#  5718
5720#  5720    5722#  5722    5724#  5724    5726#  5726    5728#  5728    5733#  5733    5735#  5735
5737#  5737    5739#  5739    5744#  5744    5746#  5746    5748#  5748    5750#  5750    5755#  5755
5757#  5757    5759#  5759    5761#  5761    5766#  5766    5768#  5768    5770#  5770    5772#  5772
5777#  5777    5779#  5779    5781#  5781    5783#  5783    5788#  5788    5790#  5790    5792#  5792
5794#  5794    5802#  5802    5804#  5804    5806#  5806    5808#  5808    5810#  5810    5812#  5812
5818#  5818    5820#  5820    5822#  5822    5824#  5824    5829#  5829    5831#  5831    6032#  6032
6034#  6034    6036#  6036    6041#  6041    6043#  6043    6045#  6045    6050#  6050    6052#  6052
6054#  6054    6059#  6059    6061#  6061    6066#  6066    6068#  6068    6070#  6070    6072#  6072
6074#  6074    6080#  6080    6082#  6082    6084#  6084    6091#  6091    6093#  6093    6095#  6095
6097#  6097    6099#  6099    6101#  6101    6103#  6103    6105#  6105    6107#  6107    6112#  6112
6114#  6114    6291#  6291    6293#  6293    6295#  6295    6300#  6300    6302#  6302    6304#  6304
6309#  6309    6311#  6311    6313#  6313    6318#  6318    6320#  6320    6325#  6325    6327#  6327
6329#  6329    6331#  6331    6333#  6333    6339#  6339    6341#  6341    6343#  6343    6351#  6351
6353#  6353    6355#  6355    6357#  6357    6359#  6359    6361#  6361    6363#  6363    6365#  6365
6367#  6367    6369#  6369    6371#  6371    6373#  6373    6378#  6378    6380#  6380    6555#  6555
6557#  6557    6559#  6559    6564#  6564    6566#  6566    6568#  6568    6573#  6573    6575#  6575
6577#  6577    6582#  6582    6584#  6584    6589#  6589    6591#  6591    6593#  6593    6595#  6595
6597#  6597    6603#  6603    6605#  6605    6607#  6607    6614#  6614    6616#  6616    6618#  6618
6620#  6620    6622#  6622    6624#  6624    6626#  6626    6628#  6628    6630#  6630    6635#  6635
6637#  6637    6826#  6826    6828#  6828    6830#  6830    6832#  6832    6834#  6834    6836#  6836
6838#  6838    6840#  6840    6955#  6955    6957#  6957    6959#  6959    6961#  6961    6963#  6963
6968#  6968    6970#  6970    6972#  6972    6974#  6974    6976#  6976    6978#  6978    6984#  6984
6986#  6986    6988#  6988    6990#  6990    6993#  6993    6995#  6995    6997#  6997    6999#  6999
7001#  7001    7003#  7003    7009#  7009    7011#  7011    7013#  7013    7015#  7015    7018#  7018
7020#  7020    7022#  7022    7024#  7024    7026#  7026    7028#  7028    7034#  7034    7036#  7036
7038#  7038    7040#  7040    7043#  7043    704[illegible]#  7045    7047#  7047    7049#  7049    7051#  7051
7053#  7053    7055#  7055    7060#  7060    706[illegible]#  7062    7381#  7381    7383#  7383    7385#  7385
7387#  7387    7389#  7389    7394#  7394    7396#  7396    7398#  7398    7400#  7400    7402#  7402
7404#  7404    7410#  7410    7412#  7412    7414#  7414    7416#  7416    7419#  7419    7421#  7421
7423#  7423    7425#  7425    7427#  7427    7429#  7429    7435#  7435    7437#  7437    7439#  7439
7441#  7441    7444#  7444    7446#  7446    7448#  7448    7450#  7450    7452#  7452    7454#  7454
7456#  7456    7461#  7461    7463#  7463    7465#  7465    7467#  7467    7469#  7469    7471#  7471
7473#  7473    7479#  7479    7481#  7481    7483#  7483    7485#  7485    7488#  7488    7490#  7490
```

.MBAD
.MCCCC
.MCCER
.MCCFZ
.MCCGC
.MCCIA
.MCENA
.MCJP
.MCJS
.MCONT
.MCRTN
.MCRY
.MD

```
7492#  7492    7494#  7494    7496#  7496    7498#  7498    7500#  7500    7505#  7505    7507#  7507
```

```
654          7492#  7492   7494#  7494   7496#  7496   7498#  7498   7500#  7500   7505#  7505   7507#  7507
     $CHR    1071#  1071   1073#  1073   1075#  1075   1077#  1077   1079#  1079   1081#  1081   1083#  1083      SEQ 0667
             1085#  1085   1087#  1087   1649#  1649   2005#  2005   2007#  2007   2009#  2009   2011#  2011
             2013#  2013   2018#  2018   2020#  2020   2022#  2022   2024#  2024   2026#  2026   2108#  2108
             2110#  2110   2112#  2112   2196#  2196   2198#  2198   2200#  2200   2202#  2202   2280#  2280
             2282#  2282   2284#  2284   2286#  2286   2363#  2363   2365#  2365   2367#  2367   2369#  2369
             2371#  2371   2373#  2373   2448#  2448   2450#  2450   2452#  2452   2532#  2532   2534#  2534
             2536#  2536   2632#  2632   2634#  2634   2636#  2636   2638#  2638   2778#  2778   2780#  2780
             2782#  2782   2784#  2784   2786#  2786   2788#  2788   2790#  2790   2792#  2792   2794#  2794
             2796#  2796   2798#  2798   2800#  2800   2802#  2802   2804#  2804   2806#  2806   2808#  2808
             2810#  2810   2812#  2812   2814#  2814   2816#  2816   2818#  2818   2820#  2820   2822#  2822
             2824#  2824   2826#  2826   2828#  2828   2830#  2830   2832#  2832   2834#  2834   2836#  2836
             2838#  2838   2840#  2840   2842#  2842   2844#  2844   2846#  2846   2848#  2848   2850#  2850             .MJ
             2852#  2852   2854#  2854   2856#  2856   2858#  2858   2860#  2860   2862#  2862   2864#  2864
             2866#  2866   2868#  2868   2870#  2870   2872#  2872   2954#  2954   2956#  2956   2958#  2958
             2960#  2960   3130#  3130   3132#  3132   3134#  3134   3136#  3136   3138#  3138   3140#  3140
             3254#  3254   3256#  3256   3258#  3258   3260#  3260   3263#  3263   3266#  3266   3268#  3268
             3270#  3270   3272#  3272   3275#  3275   3277#  3277   3279#  3279   3281#  3281   3283#  3283
             3288#  3288   3290#  3290   3292#  3292   3294#  3294   3296#  3296   3370#  3370   3372#  3372
             3374#  3374   3446#  3446   3448#  3448   3450#  3450   3526#  3526   3596#  3596   3598#  3598
             3600#  3600   3680#  3680   3804#  3804   3806#  3806   3808#  3808   3810#  3810   3815#  3815
             3817#  3817   3819#  3819   3824#  3824   3826#  3826   3831#  3831   3833#  3833   3835#  3835
             3837#  3837   3842#  3842   3844#  3844   3846#  3846   3851#  3851   3857#  3857   3859#  3859
             3864#  3864   3866#  3866   3868#  3868   3870#  3870   3875#  3875   3877#  3877   3879#  3879
             3884#  3884   3886#  3886   3891#  3891   3898#  3898   3900#  3900   3902#  3902   4025#  4025
             4027#  4027   4029#  4029   4031#  4031   4036#  4036   4038#  4038   4040#  4040   4045#  4045
             4047#  4047   4052#  4052   4054#  4054   4056#  4056   4058#  4058   4063#  4063   4065#  4065
             4067#  4067   4072#  4072   4078#  4078   4080#  4080   4085#  4085   4087#  4087   4089#  4089
             4091#  4091   4096#  4096   4098#  4098   4100#  4100   4105#  4105   4107#  4107   4112#  4112
             4119#  4119   4121#  4121   4123#  4123   4244#  4244   4246#  4246   4248#  4248   4250#  4250
             4255#  4255   4257#  4257   4259#  4259   4264#  4264   4266#  4266   4271#  4271   4273#  4273
             4275#  4275   4277#  4277   4282#  4282   4284#  4284   4286#  4286   4291#  4291   4297#  4297
             4299#  4299   4304#  4304   4306#  4306   4308#  4308   4310#  4310   4315#  4315   4317#  4317
             4319#  4319   4324#  4324   4326#  4326   4331#  4331   4338#  4338   4340#  4340   4342#  4342
             4431#  4431   4602#  4602   4604#  4604   4606#  4606   4684#  4684   4760#  4760   4885#  4885
             4890#  4890   4892#  4892   4894#  4894   4896#  4896   4898#  4898   4900#  4900   4902#  4902
             4904#  4904   4906#  4906   4908#  4908   4910#  4910   4915#  4915   4917#  4917   4919#  4919
             4921#  4921   4926#  4926   4928#  4928   4930#  4930   4932#  4932   4934#  4934   4936#  4936
             4938#  4938   4940#  4940   4942#  4942   4944#  4944   5037#  5037   5039#  5039   5041#  5041
             5164#  5164   5166#  5166   5168#  5168   5284#  5284   5286#  5286   5288#  5288   5447#  5447
             5449#  5449   5451#  5451   5453#  5453   5455#  5455   5457#  5457   5462#  5462   5464#  5464
             5466#  5466   5468#  5468   5473#  5473   5475#  5475   5477#  5477   5479#  5479   5484#  5484             .MJMAP
             5486#  5486   5488#  5488   5490#  5490   5495#  5495   5497#  5497   5499#  5499   5501#  5501
             5506#  5506   5508#  5508   5510#  5510   5512#  5512   5517#  5517   5519#  5519   5521#  5521
             5523#  5523   5531#  5531   5533#  5533   5535#  5535   5537#  5537   5539#  5539   5541#  5541
             5547#  5547   5549#  5549   5551#  5551   5553#  5553   5558#  5558   5560#  5560   5718#  5718
             5720#  5720   5722#  5722   5724#  5724   5726#  5726   5728#  5728   5733#  5733   5735#  5735
             5737#  5737   5739#  5739   5744#  5744   5746#  5746   5748#  5748   5750#  5750   5755#  5755
             5757#  5757   5759#  5759   5761#  5761   5766#  5766   5768#  5768   5770#  5770   5772#  5772
             5777#  5777   5779#  5779   5781#  5781   5783#  5783   5788#  5788   5790#  5790   5792#  5792
             5794#  5794   5802#  5802   5804#  5804   5806#  5806   5808#  5808   5810#  5810   5812#  5812
             5818#  5818   5820#  5820   5822#  5822   5824#  5824   5829#  5829   5831#  5831   6032#  6032
             6034#  6034   6036#  6036   6041#  6041   6043#  6043   6045#  6045   6050#  6050   6052#  6052
```

```
             6054#  6054   6059#  6059   6061#  6061   6066#  6066   6068#  6068   6070#  6070   6072#  6072
```

```
          6054#  6054   6059#  6059   6061#  6061   6066#  6066   6068#  6068   6070#  6070   6072#  6072     SEQ 0668
          6074#  6074   6080#  6080   6082#  6082   6084#  6084   6091#  6091   6093#  6093   6095#  6095
          6097#  6097   6099#  6099   6101#  6101   6103#  6103   6105#  6105   6107#  6107   6112#  6112
          6114#  6114   6291#  6291   6293#  6293   6295#  6295   6300#  6300   6302#  6302   6304#  6304
          6309#  6309   6311#  6311   6313#  6313   6318#  6318   6320#  6320   6325#  6325   6327#  6327
          6329#  6329   6331#  6331   6333#  6333   6339#  6339   6341#  6341   6343#  6343   6351#  6351
          6353#  6353   6355#  6355   6357#  6357   6359#  6359   6361#  6361   6363#  6363   6365#  6365
          6367#  6367   6369#  6369   6371#  6371   6373#  6373   6378#  6378   6380#  6380   6555#  6555
          6557#  6557   6559#  6559   6564#  6564   6566#  6566   6568#  6568   6573#  6573   6575#  6575
          6577#  6577   6582#  6582   6584#  6584   6589#  6589   6591#  6591   6593#  6593   6595#  6595
          6597#  6597   6603#  6603   6605#  6605   6607#  6607   6614#  6614   6616#  6616   6618#  6618
          6620#  6620   6622#  6622   6624#  6624   6626#  6626   6628#  6628   6630#  6630   6635#  6635
          6637#  6637   6826#  6826   6828#  6828   6830#  6830   6832#  6832   6834#  6834   6836#  6836
          6838#  6838   6840#  6840   6955#  6955   6957#  6957   6959#  6959   6961#  6961   6963#  6963
          6968#  6968   6970#  6970   6972#  6972   6974#  6974   6976#  6976   6978#  6978   6984#  6984
          6986#  6986   6988#  6988   6990#  6990   6993#  6993   6995#  6995   6997#  6997   6999#  6999
          7001#  7001   7003#  7003   7009#  7009   7011#  7011   7013#  7013   7015#  7015   7018#  7018
          7020#  7020   7022#  7022   7024#  7024   7026#  7026   7028#  7028   7034#  7034   7036#  7036
          7038#  7038   7040#  7040   7043#  7043   7045#  7045   7047#  7047   7049#  7049   7051#  7051
          7053#  7053   7055#  7055   7060#  7060   7062#  7062   7381#  7381   7383#  7383   7385#  7385
          7387#  7387   7389#  7389   7394#  7394   7396#  7396   7398#  7398   7400#  7400   7402#  7402
          7404#  7404   7410#  7410   7412#  7412   7414#  7414   7416#  7416   7419#  7419   7421#  7421
          7423#  7423   7425#  7425   7427#  7427   7429#  7429   7435#  7435   7437#  7437   7439#  7439
          7441#  7441   7444#  7444   7446#  7446   7448#  7448   7450#  7450   7452#  7452   7454#  7454
          7456#  7456   7461#  7461   7463#  7463   7465#  7465   7467#  7467   7469#  7469   7471#  7471
          7473#  7473   7479#  7479   7481#  7481   7483#  7483   7485#  7485   7488#  7488   7490#  7490
          7492#  7492   7494#  7494   7496#  7496   7498#  7498   7500#  7500   7505#  7505   7507#  7507
$GARG     1071#  1071   1073#  1073   1075#  1075   1077#  1077   1079#  1079   1081#  1081   1083#  1083
          1085#  1085   1087#  1087   1649#  1649   2005#  2005   2007#  2007   2009#  2009   2011#  2011
          2013#  2013   2018#  2018   2020#  2020   2022#  2022   2024#  2024   2026#  2026   2108#  2108
          2110#  2110   2112#  2112   2196#  2196   2198#  2198   2200#  2200   2202#  2202   2280#  2280
          2282#  2282   2284#  2284   2286#  2286   2363#  2363   2365#  2365   2367#  2367   2369#  2369
          2371#  2371   2373#  2373   2448#  2448   2450#  2450   2452#  2452   2532#  2532   2534#  2534
          2536#  2536   2632#  2632   2634#  2634   2636#  2636   2638#  2638   2778#  2778   2780#  2780
          2782#  2782   2784#  2784   2786#  2786   2788#  2788   2790#  2790   2792#  2792   2794#  2794
          2796#  2796   2798#  2798   2800#  2800   2802#  2802   2804#  2804   2806#  2806   2808#  2808
          2810#  2810   2812#  2812   2814#  2814   2816#  2816   2818#  2818   2820#  2820   2822#  2822
          2824#  2824   2826#  2826   2828#  2828   2830#  2830   2832#  2832   2834#  2834   2836#  2836
          2838#  2838   2840#  2840   2842#  2842   2844#  2844   2846#  2846   2848#  2848   2850#  2850
          2852#  2852   2854#  2854   2856#  2856   2858#  2858   2860#  2860   2862#  2862   2864#  2864
          2866#  2866   2868#  2868   2870#  2870   2872#  2872   2954#  2954   2956#  2956   2958#  2958
          2960#  2960   3130#  3130   3132#  3132   3134#  3134   3136#  3136   3138#  3138   3140#  3140
          3254#  3254   3256#  3256   3258#  3258   3260#  3260   3263#  3263   3266#  3266   3268#  3268
          3270#  3270   3272#  3272   3275#  3275   3277#  3277   3279#  3279   3281#  3281   3283#  3283
          3288#  3288   3290#  3290   3292#  3292   3294#  3294   3296#  3296   3370#  3370   3372#  3372
          3374#  3374   3446#  3446   3448#  3448   3450#  3450   3526#  3526   3596#  3596   3598#  3598
          3600#  3600   3680#  3680   3804#  3804   3806#  3806   3808#  3808   3810#  3810   3815#  3615
          3817#  3817   3819#  3819   3824#  3824   3826#  3826   3831#  3831   3833#  3833   3835#  3835
          3837#  3837   3842#  3842   3844#  3844   3846#  3846   3851#  3851   3857#  3857   3859#  3859
          3864#  3864   3866#  3866   3868#  3868   3870#  3870   3875#  3875   3877#  3877   3879#  3879
          3884#  3884   3886#  3886   3891#  3891   3898#  3898   3900#  3900   3902#  3902   4025#  4025
          4027#  4027   4029#  4029   4031#  4031   4036#  4036   4038#  4038   4040#  4040   4045#  4045
          4047#  4047   4052#  4052   4054#  4054   4056#  4056   4058#  4058   4063#  4063   4065#  4065
```

```
4067#  4067   4072#  4072   4078#  4078   4080#  4080   4085#  4085   4087#  4087   4089#  4089
4091#  4091   4096#  4096   4098#  4098   4100#  4100   4105#  4105   4107#  4107   4112#  4112      SEQ 0669
4119#  4119   4121#  4121   4123#  4123   4244#  4244   4246#  4246   4248#  4248   4250#  4250
4255#  4255   4257#  4257   4259#  4259   4264#  4264   4266#  4266   4271#  4271   4273#  4273
4275#  4275   4277#  4277   4282#  4282   4284#  4284   4286#  4286   4291#  4291   4297#  4297
4299#  4299   4304#  4304   4306#  4306   4308#  4308   4310#  4310   4315#  4315   4317#  4317
4319#  4319   4324#  4324   4326#  4326   4331#  4331   4338#  4338   4340#  4340   4342#  4342
4431#  4431   4602#  4602   4604#  4604   4606#  4606   4684#  4684   4760#  4760   4885#  4885
4890#  4890   4892#  4892   4894#  4894   4896#  4896   4898#  4898   4900#  4900   4902#  4902
4904#  4904   4906#  4906   4908#  4908   4910#  4910   4915#  4915   4917#  4917   4919#  4919
4921#  4921   4926#  4926   4928#  4928   4930#  4930   4932#  4932   4934#  4934   4936#  4936
4938#  4938   4940#  4940   4942#  4942   4944#  4944   5037#  5037   5039#  5039   5041#  5041
5164#  5164   5166#  5166   5168#  5168   5284#  5284   5286#  5286   5288#  5288   5447#  5447
5449#  5449   5451#  5451   5453#  5453   5455#  5455   5457#  5457   5462#  5462   5464#  5464
5466#  5466   5468#  5468   5473#  5473   5475#  5475   5477#  5477   5479#  5479   5484#  5484
5486#  5486   5488#  5488   5490#  5490   5495#  5495   5497#  5497   5499#  5499   5501#  5501
5506#  5506   5508#  5508   5510#  5510   5512#  5512   5517#  5517   5519#  5519   5521#  5521
5523#  5523   5531#  5531   5533#  5533   5535#  5535   5537#  5537   5539#  5539   5541#  5541
5547#  5547   5549#  5549   5551#  5551   5553#  5553   5558#  5558   5560#  5560   5718#  5718
5720#  5720   5722#  5722   5724#  5724   5726#  5726   5728#  5728   5733#  5733   5735#  5735
5737#  5737   5739#  5739   5744#  5744   5746#  5746   5748#  5748   5750#  5750   5755#  5755
5757#  5757   5759#  5759   5761#  5761   5766#  5766   5768#  5768   5770#  5770   5772#  5772
5777#  5777   5779#  5779   5781#  5781   5783#  5783   5788#  5788   5790#  5790   5792#  5792
5794#  5794   5802#  5802   5804#  5804   5806#  5806   5808#  5808   5810#  5810   5812#  5812
5818#  5818   5820#  5820   5822#  5822   5824#  5824   5829#  5829   5831#  5831   6032#  6032
6034#  6034   6036#  6036   6041#  6041   6043#  6043   6045#  6045   6050#  6050   6052#  6052
6054#  6054   6059#  6059   6061#  6061   6066#  6066   6068#  6068   6070#  6070   6072#  6072
6074#  6074   6080#  6080   6082#  6082   6084#  6084   6091#  6091   6093#  6093   6095#  6095
6097#  6097   6099#  6099   6101#  6101   6103#  6103   6105#  6105   6107#  6107   6112#  6112
6114#  6114   6291#  6291   6293#  6293   6295#  6295   6300#  6300   6302#  6302   6304#  6304
6309#  6309   6311#  6311   6313#  6313   6318#  6318   6320#  6320   6325#  6325   6327#  6327
6329#  6329   6331#  6331   6333#  6333   6339#  6339   6341#  6341   6343#  6343   6351#  6351
6353#  6353   6355#  6355   6357#  6357   6359#  6359   6361#  6361   6363#  6363   6365#  6365
6367#  6367   6369#  6369   6371#  6371   6373#  6373   6378#  6378   6380#  6380   6555#  6555
6557#  6557   6559#  6559   6564#  6564   6566#  6566   6568#  6568   6573#  6573   6575#  6575
6577#  6577   6582#  6582   6584#  6584   6589#  6589   6591#  6591   6593#  6593   6595#  6595
6597#  6597   6603#  6603   6605#  6605   6607#  6607   6614#  6614   6616#  6616   6618#  6618
6620#  6620   6622#  6622   6624#  6624   6626#  6626   6628#  6628   6630#  6630   6635#  6635
6637#  6637   6826#  6826   6828#  6828   6830#  6830   6832#  6832   6834#  6834   6836#  6836
6838#  6838   6840#  6840   6955#  6955   6957#  6957   6959#  6959   6961#  6961   6963#  6963
6968#  6968   6970#  6970   6972#  6972   6974#  6974   6976#  6976   6978#  6978   6984#  6984
6986#  6986   6988#  6988   6990#  6990   6993#  6993   6995#  6995   6997#  6997   6999#  6999
7001#  7001   7003#  7003   7009#  7009   7011#  7011   7013#  7013   7015#  7015   7018#  7018
7020#  7020   7022#  7022   7024#  7024   7026#  7026   7028#  7028   7034#  7034   7036#  7036
7038#  7038   7040#  7040   7043#  7043   7045#  7045   7047#  7047   7049#  7049   7051#  7051
7053#  7053   7055#  7055   7060#  7060   7062#  7062   7381#  7381   7383#  7383   7385#  7385
7387#  7387   7389#  7389   7394#  7394   7396#  7396   7398#  7398   7400#  7400   7402#  7402
7404#  7404   7410#  7410   7412#  7412   7414#  7414   7416#  7416   7419#  7419   7421#  7421
7423#  7423   7425#  7425   7427#  7427   7429#  7429   7435#  7435   7437#  7437   7439#  7439
7441#  7441   7444#  7444   7446#  7446   7448#  7448   7450#  7450   7452#  7452   7454#  7454
7456#  7456   7461#  7461   7463#  7463   7465#  7465   7467#  7467   7469#  7469   7471#  7471
7473#  7473   7479#  7479   7481#  7481   7483#  7483   7485#  7485   7488#  7488   7490#  7490
7492#  7492   7494#  7494   7496#  7496   7498#  7498   7500#  7500   7505#  7505   7507#  7507
```

```
%ADDR    1071#  1071   1073   1073#  1075   1075#  1077   1077#  1079   1079#  1081   1081#  1083   1083#        SEQ 0670
         1085   1085#  1087   1087#  1649#  1649   2005#  2005   2007   2007#  2009   2009#  2011   2011#
         2013   2013#  2018   2018#  2020   2020#  2022   2022#  2024   2024#  2026   2026#  2108#  2108
         2110   2110#  2112   2112#  2196#  2196   2198   2198#  2200   2200#  2202   2202#  2280#  2280
         2282   2282#  2284   2284#  2286   2286#  2363#  2363   2365   2365#  2367   2367#  2369   2369#
         2371   2371#  2373   2373#  2448#  2448   2450   2450#  2452   2452#  2532#  2532   2534   2534#
         2536   2536#  2632#  2632   2634   2634#  2636   2636#  2638   2638#  2778#  2778   2780   2780#
         2782   2782#  2784   2784#  2786   2786#  2788   2788#  2790   2790#  2792   2792#  2794   2794#
         2796   2796#  2798   2798#  2800   2800#  2802   2802#  2804   2804#  2806   2806#  2808   2808#
         2810   2810#  2812   2812#  2814   2814#  2816   2816#  2818   2818#  2820   2820#  2822   2822#
         2824   2824#  2826   2826#  2828   2828#  2830   2830#  2832   2832#  2834   2834#  2836   2836#
         2838   2838#  2840   2840#  2842   2842#  2844   2844#  2846   2846#  2848   2848#  2850   2850#
         2852   2852#  2854   2854#  2856   2856#  2858   2858#  2860   2860#  2862   2862#  2864   2864#
         2866   2866#  2868   2868#  2870   2870#  2872   2872#  2954#  2954   2956   2956#  2958   2958#
         2960   2960#  3130#  3130   3132   3132#  3134   3134#  3136   3136#  3138   3138#  3140   3140#
         3254#  3254   3256   3256#  3258   3258#  3260   3260#  3263#  3263   3266#  3266   3268   3268#
         3270   3270#  3272   3272#  3275#  3275   3277   3277#  3279   3279#  3281   3281#  3283   3283#
         3288#  3288   3290   3290#  3292   3292#  3294   3294#  3296   3296#  3370#  3370   3372   3372#
         3374   3374#  3446#  3446   3448   3448#  3450   3450#  3526#  3526   3596#  3596   3598   3598#
         3600   3600#  3680#  3680   3804#  3804   3806   3806#  3808   3808#  3810   3810#  3815   3815#
         3817   3817#  3819   3819#  3824   3824#  3826   3826#  3831   3831#  3833   3833#  3835   3835#
         3837   3837#  3842   3842#  3844   3844#  3846   3846#  3851   3851#  3857   3857#  3859   3859#
         3864   3864#  3866   3866#  3868   3868#  3870   3870#  3875   3875#  3877   3877#  3879   3879#
         3884   3884#  3886   3886#  3891   3891#  3898#  3898   3900#  3900   3902#  3902   4025#  4025
         4027   4027#  4029   4029#  4031   4031#  4036   4036#  4038   4038#  4040   4040#  4045   4045#
         4047   4047#  4052   4052#  4054   4054#  4056   4056#  4058   4058#  4063   4063#  4065   4065#
         4067   4067#  4072   4072#  4078   4078#  4080   4080#  4085   4085#  4087   4087#  4089   4089#
         4091   4091#  4096   4096#  4098   4098#  4100   4100#  4105   4105#  4107   4107#  4112   4112#
         4119#  4119   4121#  4121   4123#  4123   4244#  4244   4246   4246#  4248   4248#  4250   4250#
         4255   4255#  4257   4257#  4259   4259#  4264   4264#  4266   4266#  4271   4271#  4273   4273#
         4275   4275#  4277   4277#  4282   4282#  4284   4284#  4286   4286#  4291   4291#  4297   4297#
         4299   4299#  4304   4304#  4306   4306#  4308   4308#  4310   4310#  4315   4315#  4317   4317#
         4319   4319#  4324   4324#  4326   4326#  4331   4331#  4338#  4338   4340#  4340   4342#  4342
         4431#  4431   4602#  4602   4604   4604#  4606   4606#  4684#  4684   4760#  4760   4885#  4885
         4890#  4890   4892   4892#  4894   4894#  4896   4896#  4898   4898#  4900   4900#  4902   4902#
         4904   4904#  4906   4906#  4908   4908#  4910   4910#  4915#  4915   4917   4917#  4919   4919#
         4921   4921#  4926#  4926   4928   4928#  4930   4930#  4932   4932#  4934   4934#  4936   4936#
         4938   4938#  4940   4940#  4942   4942#  4944   4944#  5037#  5037   5039   5039#  5041   5041#
         5164#  5164   5166   5166#  5168   5168#  5284#  5284   5286   5286#  5288   5288#  5447#  5447
         5449   5449#  5451   5451#  5453   5453#  5455   5455#  5457   5457#  5462   5462#  5464   5464#
         5466   5466#  5468   5468#  5473   5473#  5475   5475#  5477   5477#  5479   5479#  5484   5484#
         5486   5486#  5488   5488#  5490   5490#  5495   5495#  5497   5497#  5499   5499#  5501   5501#
         5506   5506#  5508   5508#  5510   5510#  5512   5512#  5517   5517#  5519   5519#  5521   5521#
         5523   5523#  5531#  5531   5533   5533#  5535   5535#  5537   5537#  5539   5539#  5541   5541#
         5547#  5547   5549   5549#  5551   5551#  5553   5553#  5558#  5558   5560   5560#  5718#  5718
         5720   5720#  5722   5722#  5724   5724#  5726   5726#  5728   5728#  5733   5733#  5735   5735#
         5737   5737#  5739   5739#  5744   5744#  5746   5746#  5748   5748#  5750   5750#  5755   5755#
         5757   5757#  5759   5759#  5761   5761#  5766   5766#  5768   5768#  5770   5770#  5772   5772#
         5777   5777#  5779   5779#  5781   5781#  5783   5783#  5788   5788#  5790   5790#  5792   5792#
         5794   5794#  5802#  5802   5804   5804#  5806   5806#  5808   5808#  5810   5810#  5812   5812#
         5818#  5818   5820   5820#  5822   5822#  5824   5824#  5829#  5829   5831   5831#  6032#  6032
         6034   6034#  6036   6036#  6041   6041#  6043   6043#  6045   6045#  6050   6050#  6052   6052#
         6054   6054#  6059#  6059   6061   6061#  6066#  6066   6068   6068#  6070   6070#  6072   6072#
```

```
          6074      6074#     6080#     6080      6082      6082#     6084      6084#     6091#     6091      6093      6093#     6095      6095#
          6097      6097#     6099      6099#     6101      6101#     6103      6103#     6105      6105#     6107      6107#     6112#     6112        SEQ 0671
          6114      6114#     6291#     6291      6293      6293#     6295      6295#     6300      6300#     6302      6302#     6304      6304#
          6309      6309#     6311      6311#     6313      6313#     6318#     6318      6320      6320#     6325#     6325      6327      6327#
          6329      6329#     6331      6331#     6333      6333#     6339#     6339      6341      6341#     6343      6343#     6351#     6351
          6353      6353#     6355      6355#     6357      6357#     6359      6359#     6361      6361#     6363      6363#     6365      6365#
          6367      6367#     6369      6369#     6371      6371#     6373      6373#     6378#     6378      6380      6380#     6555#     6555
          6557      6557#     6559      6559#     6564      6564#     6566      6566#     6568      6568#     6573      6573#     6575      6575#
          6577      6577#     6582#     6582      6584      6584#     6589#     6589      6591      6591#     6593      6593#     6595      6595#
          6597      6597#     6603#     6603      6605      6605#     6607      6607#     6614#     6614      6616      6616#     6618      6618#
          6620      6620#     6622      6622#     6624      6624#     6626      6626#     6628      6628#     6630      6630#     6635#     6635
          6637      6637#     6826#     6826      6828      6828#     6830      6830#     6832      6832#     6834      6834#     6836      6836#
          6838      6838#     6840      6840#     6955#     6955      6957      6957#     6959      6959#     6961      6961#     6963      6963#
          6968      6968#     6970      6970#     6972      6972#     6974      6974#     6976      6976#     6978      6978#     6984      6984#
          6986      6986#     6988      6988#     6990      6990#     6993      6993#     6995      6995#     6997      6997#     6999      6999#
          7001      7001#     7003      7003#     7009      7009#     7011      7011#     7013      7013#     7015      7015#     7018      7018#
          7020      7020#     7022      7022#     7024      7024#     7026      7026#     7028      7028#     7034      7034#     7036      7036#
          7038      7038#     7040      7040#     7043      7043#     7045      7045#     7047      7047#     7049      7049#     7051      7051#
          7053      7053#     7055      7055#     7060#     7060      7062      7062#     7381#     7381      7383      7383#     7385      7385#
          7387      7387#     7389      7389#     7394      7394#     7396      7396#     7398      7398#     7400      7400#     7402      7402#
          7404      7404#     7410      7410#     7412      7412#     7414      7414#     7416      7416#     7419      7419#     7421      7421#
          7423      7423#     7425      7425#     7427      7427#     7429      7429#     7435      7435#     7437      7437#     7439      7439#
          7441      7441#     7444      7444#     7446      7446#     7448      7448#     7450      7450#     7452      7452#     7454      7454#
          7456      7456#     7461#     7461      7463      7463#     7465      7465#     7467      7467#     7469      7469#     7471      7471#
          7473      7473#     7479      7479#     7481      7481#     7483      7483#     7485      7485#     7488      7488#     7490      7490#
          7492      7492#     7494      7494#     7496      7496#     7498      7498#     7500      7500#     7505#     7505      7507      7507#
%ML       1071#     1071      1073#     1073      1075#     1075      1077#     1077      1079#     1079      1081#     1081      1083#     1083
          1085#     1085      1087#     1087      1649#     1649      2005#     2005      2007#     2007      2009#     2009      2011#     2011
          2013#     2013      2018#     2018      2020#     2020      2022#     2022      2024#     2024      2026#     2026      2108#     2108
          2110#     2110      2112#     2112      2196#     2196      2198#     2198      2200#     2200      2202#     2202      2280#     2280
          2282#     2282      2284#     2284      2286#     2286      2363#     2363      2365#     2365      2367#     2367      2369#     2369
          2371#     2371      2373#     2373      2448#     2448      2450#     2450      2452#     2452      2532#     2532      2534#     2534
          2536#     2536      2632#     2632      2634#     2634      2636#     2636      2638#     2638      2778#     2778      2780#     2780
          2782#     2782      2784#     2784      2786#     2786      2788#     2788      2790#     2790      2792#     2792      2794#     2794
          2796#     2796      2798#     2798      2800#     2800      2802#     2802      2804#     2804      2806#     2806      2808#     2808
          2810#     2810      2812#     2812      2814#     2814      2816#     2816      2818#     2818      2820#     2820      2822#     2822
          2824#     2824      2826#     2826      2828#     2828      2830#     2830      2832#     2832      2834#     2834      2836#     2836
          2838#     2838      2840#     2840      2842#     2842      2844#     2844      2846#     2846      2848#     2848      2850#     2850
          2852#     2852      2854#     2854      2856#     2856      2858#     2858      2860#     2860      2862#     2862      2864#     2864
          2866#     2866      2868#     2868      2870#     2870      2872#     2872      2954#     2954      2956#     2956      2958#     2958
          2960#     2960      3130#     3130      3132#     3132      3134#     3134      3136#     3136      3138#     3138      3140#     3140
          3254#     3254      3256#     3256      3258#     3258      3260#     3260      3263#     3263      3266#     3266      3268#     3268
          3270#     3270      3272#     3272      3275#     3275      3277#     3277      3279#     3279      3281#     3281      3283#     3283
          3288#     3288      3290#     3290      3292#     3292      3294#     3294      3296#     3296      3370#     3370      3372#     3372
          3374#     3374      3446#     3446      3448#     3448      3450#     3450      3526#     3526      3596#     3596      3598#     3598
          3600#     3600      3680#     3680      3804#     3804      3806#     3806      3808#     3808      3810#     3810      3815#     3815
          3817#     3817      3819#     3819      3824#     3824      3826#     3826      3831#     3831      3833#     3833      3835#     3835
          3837#     3837      3842#     3842      3844#     3844      3846#     3846      3851#     3851      3857#     3857      3859#     3859
          3864#     3864      3866#     3866      3868#     3868      3870#     3870      3875#     3875      3877#     3877      3879#     3879
          3884#     3884      3886#     3886      3891#     3891      3898#     3898      3900#     3900      3902#     3902      4025#     4025
          4027#     4027      4029#     4029      4031#     4031      4036#     4036      4038#     4038      4040#     4040      4045#     4045
          4047#     4047      4052#     4052      4054#     4054      4056#     4056      4058#     4058      4063#     4063      4065#     4065
          4067#     4067      4072#     4072      4078#     4078      4080#     4080      4085#     4085      4087#     4087      4089#     4089
          4091#  4091    4096#  4096    4098#  4098    4100#  4100    4105#  4105    4107#  4107    4112#  4112
          4119#  4119    4121#  4121    4123#  4123    4244#  4244    4246#  4246    4248#  4248    4250#  4250        SEQ 0672
          4255#  4255    4257#  4257    4259#  4259    4264#  4264    4266#  4266    4271#  4271    4273#  4273
          4275#  4275    4277#  4277    4282#  4282    4284#  4284    4286#  4286    4291#  4291    4297#  4297
          4299#  4299    4304#  4304    4306#  4306    4308#  4308    4310#  4310    4315#  4315    4317#  4317
          4319#  4319    4324#  4324    4326#  4326    4331#  4331    4338#  4338    4340#  4340    4342#  4342
          4431#  4431    4602#  4602    4604#  4604    4606#  4606    4684#  4684    4760#  4760    4885#  4885
          4890#  4890    4892#  4892    4894#  4894    4896#  4896    4898#  4898    4900#  4900    4902#  4902
          4904#  4904    4906#  4906    4908#  4908    4910#  4910    4915#  4915    4917#  4917    4919#  4919
          4921#  4921    4926#  4926    4928#  4928    4930#  4930    4932#  4932    4934#  4934    4936#  4936
          4938#  4938    4940#  4940    4942#  4942    4944#  4944    5037#  5037    5039#  5039    5041#  5041
          5164#  5164    5166#  5166    5168#  5168    5284#  5284    5286#  5286    5288#  5288    5447#  5447
          5449#  5449    5451#  5451    5453#  5453    5455#  5455    5457#  5457    5462#  5462    5464#  5464
          5466#  5466    5468#  5468    5473#  5473    5475#  5475    5477#  5477    5479#  5479    5484#  5484
          5486#  5486    5488#  5488    5490#  5490    5495#  5495    5497#  5497    5499#  5499    5501#  5501
          5506#  5506    5508#  5508    5510#  5510    5512#  5512    5517#  5517    5519#  5519    5521#  5521
          5523#  5523    5531#  5531    5533#  5533    5535#  5535    5537#  5537    5539#  5539    5541#  5541
          5547#  5547    5549#  5549    5551#  5551    5553#  5553    5558#  5558    5560#  5560    5718#  5718
          5720#  5720    5722#  5722    5724#  5724    5726#  5726    5728#  5728    5733#  5733    5735#  5735
          5737#  5737    5739#  5739    5744#  5744    5746#  5746    5748#  5748    5750#  5750    5755#  5755
          5757#  5757    5759#  5759    5761#  5761    5766#  5766    5768#  5768    5770#  5770    5772#  5772
          5777#  5777    5779#  5779    5781#  5781    5783#  5783    5788#  5788    5790#  5790    5792#  5792
          5794#  5794    5802#  5802    5804#  5804    5806#  5806    5808#  5808    5810#  5810    5812#  5812
          5818#  5818    5820#  5820    5822#  5822    5824#  5824    5829#  5829    5831#  5831    6032#  6032
          6034#  6034    6036#  6036    6041#  6041    6043#  6043    6045#  6045    6050#  6050    6052#  6052
          6054#  6054    6059#  6059    6061#  6061    6066#  6066    6068#  6068    6070#  6070    6072#  6072
          6074#  6074    6080#  6080    6082#  6082    6084#  6084    6091#  6091    6093#  6093    6095#  6095
          6097#  6097    6099#  6099    6101#  6101    6103#  6103    6105#  6105    6107#  6107    6112#  6112
          6114#  6114    6291#  6291    6293#  6293    6295#  6295    6300#  6300    6302#  6302    6304#  6304
          6309#  6309    6311#  6311    6313#  6313    6318#  6318    6320#  6320    6325#  6325    6327#  6327
          6329#  6329    6331#  6331    6333#  6333    6339#  6339    6341#  6341    6343#  6343    6351#  6351
          6353#  6353    6355#  6355    6357#  6357    6359#  6359    6361#  6361    6363#  6363    6365#  6365
          6367#  6367    6369#  6369    6371#  6371    6373#  6373    6378#  6378    6380#  6380    6555#  6555
          6557#  6557    6559#  6559    6564#  6564    6566#  6566    6568#  6568    6573#  6573    6575#  6575
          6577#  6577    6582#  6582    6584#  6584    6589#  6589    6591#  6591    6593#  6593    6595#  6595
          6597#  6597    6603#  6603    6605#  6605    6607#  6607    6614#  6614    6616#  6616    6618#  6618
          6620#  6620    6622#  6622    6624#  6624    6626#  6626    6628#  6628    6630#  6630    6635#  6635
          6637#  6637    6826#  6826    6828#  6828    6830#  6830    6832#  6832    6834#  6834    6836#  6836
          6838#  6838    6840#  6840    6955#  6955    6957#  6957    6959#  6959    6961#  6961    6963#  6963
          6968#  6968    6970#  6970    6972#  6972    6974#  6974    6976#  6976    6978#  6978    6984#  6984
          6986#  6986    6988#  6988    6990#  6990    6993#  6993    6995#  6995    6997#  6997    6999#  6999
          7001#  7001    7003#  7003    7009#  7009    7011#  7011    7013#  7013    7015#  7015    7018#  7018
          7020#  7020    7022#  7022    7024#  7024    7026#  7026    7028#  7028    7034#  7034    7036#  7036
          7038#  7038    7040#  7040    7043#  7043    7045#  7045    7047#  7047    7049#  7049    7051#  7051
          7053#  7053    7055#  7055    7060#  7060    7062#  7062    7381#  7381    7383#  7383    7385#  7385
          7387#  7387    7389#  7389    7394#  7394    7396#  7396    7398#  7398    7400#  7400    7402#  7402
          7404#  7404    7410#  7410    7412#  7412    7414#  7414    7416#  7416    7419#  7419    7421#  7421
          7423#  7423    7425#  7425    7427#  7427    7429#  7429    7435#  7435    7437#  7437    7439#  7439
          7441#  7441    7444#  7444    7446#  7446    7448#  7448    7450#  7450    7452#  7452    7454#  7454
          7456#  7456    7461#  7461    7463#  7463    7465#  7465    7467#  7467    7469#  7469    7471#  7471
          7473#  7473    7479#  7479    7481#  7481    7483#  7483    7485#  7485    7488#  7488    7490#  7490
          7492#  7492    7494#  7494    7496#  7496    7498#  7498    7500#  7500    7505#  7505    7507#  7507
%MR       1071#  1071    1073#  1073    1075#  1075    1077#  1077    1079#  1079    1081#  1081    1083#  1083
```

```
1085#   1085    1087#   1087    1649#   1649    2005#   2005    2007#   2007    2009#   2009    2011#   2011
2013#   2013    2018#   2018    2020#   2020    2022#   2022    2024#   2024    2026#   2026    2108#   2108        SEQ 0673
2110#   2110    2112#   2112    2196#   2196    2198#   2198    2200#   2200    2202#   2202    2280#   2280
2282#   2282    2284#   2284    2286#   2286    2363#   2363    2365#   2365    2367#   2367    2369#   2369
2371#   2371    2373#   2373    2448#   2448    2450#   2450    2452#   2452    2532#   2532    2534#   2534
2536#   2536    2632#   2632    2634#   2634    2636#   2636    2638#   2638    2778#   2778    2780#   2780
2782#   2782    2784#   2784    2786#   2786    2788#   2788    2790#   2790    2792#   2792    2794#   2794
2796#   2796    2798#   2798    2800#   2800    2802#   2802    2804#   2804    2806#   2806    2808#   2808
2810#   2810    2812#   2812    2814#   2814    2816#   2816    2818#   2818    2820#   2820    2822#   2822
2824#   2824    2826#   2826    2828#   2828    2830#   2830    2832#   2832    2834#   2834    2836#   2836
2838#   2838    2840#   2840    2842#   2842    2844#   2844    2846#   2846    2848#   2848    2850#   2850
2852#   2852    2854#   2854    2856#   2856    2858#   2858    2860#   2860    2862#   2862    2864#   2864
2866#   2866    2868#   2868    2870#   2870    2872#   2872    2954#   2954    2956#   2956    2958#   2958
2960#   2960    3130#   3130    3132#   3132    3134#   3134    3136#   3136    3138#   3138    3140#   3140
3254#   3254    3256#   3256    3258#   3258    3260#   3260    3263#   3263    3266#   3266    3268#   3268
3270#   3270    3272#   3272    3275#   3275    3277#   3277    3279#   3279    3281#   3281    3283#   3283
3288#   3288    3290#   3290    3292#   3292    3294#   3294    3296#   3296    3370#   3370    3372#   3372
3374#   3374    3446#   3446    3448#   3448    3450#   3450    3526#   3526    3596#   3596    3598#   3598
3600#   3600    3680#   3680    3804#   3804    3806#   3806    3808#   3808    3810#   3810    3815#   3815
3817#   3817    3819#   3819    3824#   3824    3826#   3826    3831#   3831    3833#   3833    3835#   3835
3837#   3837    3842#   3842    3844#   3844    3846#   3846    3851#   3851    3857#   3857    3859#   3859
3864#   3864    3866#   3866    3868#   3868    3870#   3870    3875#   3875    3877#   3877    3879#   3879
3884#   3884    3886#   3886    3891#   3891    3898#   3898    3900#   3900    3902#   3902    4025#   4025
4027#   4027    4029#   4029    4031#   4031    4036#   4036    4038#   4038    4040#   4040    4045#   4045
4047#   4047    4052#   4052    4054#   4054    4056#   4056    4058#   4058    4063#   4063    4065#   4065
4067#   4067    4072#   4072    4078#   4078    4080#   4080    4085#   4085    4087#   4087    4089#   4089
4091#   4091    4096#   4096    4098#   4098    4100#   4100    4105#   4105    4107#   4107    4112#   4112
4119#   4119    4121#   4121    4123#   4123    4244#   4244    4246#   4246    4248#   4248    4250#   4250
4255#   4255    4257#   4257    4259#   4259    4264#   4264    4266#   4266    4271#   4271    4273#   4273
4275#   4275    4277#   4277    4282#   4282    4284#   4284    4286#   4286    4291#   4291    4297#   4297
4299#   4299    4304#   4304    4306#   4306    4308#   4308    4310#   4310    4315#   4315    4317#   4317
4319#   4319    4324#   4324    4326#   4326    4331#   4331    4338#   4338    4340#   4340    4342#   4342
4431#   4431    4602#   4602    4604#   4604    4606#   4606    4684#   4684    4760#   4760    4885#   4885
4890#   4890    4892#   4892    4894#   4894    4896#   4896    4898#   4898    4900#   4900    4902#   4902
4904#   4904    4906#   4906    4908#   4908    4910#   4910    4915#   4915    4917#   4917    4919#   4919
4921#   4921    4926#   4926    4928#   4928    4930#   4930    4932#   4932    4934#   4934    4936#   4936
4938#   4938    4940#   4940    4942#   4942    4944#   4944    5037#   5037    5039#   5039    5041#   5041
5164#   5164    5166#   5166    5168#   5168    5284#   5284    5286#   5286    5288#   5288    5447#   5447
5449#   5449    5451#   5451    5453#   5453    5455#   5455    5457#   5457    5462#   5462    5464#   5464
5466#   5466    5468#   5468    5473#   5473    5475#   5475    5477#   5477    5479#   5479    5484#   5484
5486#   5486    5488#   5488    5490#   5490    5495#   5495    5497#   5497    5499#   5499    5501#   5501
5506#   5506    5508#   5508    5510#   5510    5512#   5512    5517#   5517    5519#   5519    5521#   5521
5523#   5523    5531#   5531    5533#   5533    5535#   5535    5537#   5537    5539#   5539    5541#   5541
5547#   5547    5549#   5549    5551#   5551    5553#   5553    5558#   5558    5560#   5560    5718#   5718
5720#   5720    5722#   5722    5724#   5724    5726#   5726    5728#   5728    5733#   5733    5735#   5735
5737#   5737    5739#   5739    5744#   5744    5746#   5746    5748#   5748    5750#   5750    5755#   5755
5757#   5757    5759#   5759    5761#   5761    5766#   5766    5768#   5768    5770#   5770    5772#   5772
5777#   5777    5779#   5779    5781#   5781    5783#   5783    5788#   5788    5790#   5790    5792#   5792
5794#   5794    5802#   5802    5804#   5804    5806#   5806    5808#   5808    5810#   5810    5812#   5812
5818#   5818    5820#   5820    5822#   5822    5824#   5824    5829#   5829    5831#   5831    6032#   6032
6034#   6034    6036#   6036    6041#   6041    6043#   6043    6045#   6045    6050#   6050    6052#   6052
6054#   6054    6059#   6059    6061#   6061    6066#   6066    6068#   6068    6070#   6070    6072#   6072
6074#   6074    6080#   6080    6082#   6082    6084#   6084    6091#   6091    6093#   6093    6095#   6095
```

```
         6097#  6097   6099#  6099   6101#  6101   6103#  6103   6105#  6105   6107#  6107   6112#  6112
         6114#  6114   6291#  6291   6293#  6293   6295#  6295   6300#  6300   6302#  6302   6304#  6304
         6309#  6309   6311#  6311   6313#  6313   6318#  6318   6320#  6320   6325#  6325   6327#  6327
         6329#  6329   6331#  6331   6333#  6333   6339#  6339   6341#  6341   6343#  6343   6351#  6351
         6353#  6353   6355#  6355   6357#  6357   6359#  6359   6361#  6361   6363#  6363   6365#  6365
         6367#  6367   6369#  6369   6371#  6371   6373#  6373   6378#  6378   6380#  6380   6555#  6555
         6557#  6557   6559#  6559   6564#  6564   6566#  6566   6568#  6568   6573#  6573   6575#  6575
         6577#  6577   6582#  6582   6584#  6584   6589#  6589   6591#  6591   6593#  6593   6595#  6595
         6597#  6597   6603#  6603   6605#  6605   6607#  6607   6614#  6614   6616#  6616   6618#  6618
         6620#  6620   6622#  6622   6624#  6624   6626#  6626   6628#  6628   6630#  6630   6635#  6635
         6637#  6637   6826#  6826   6828#  6828   6830#  6830   6832#  6832   6834#  6834   6836#  6836
         6838#  6838   6840#  6840   6955#  6955   6957#  6957   6959#  6959   6961#  6961   6963#  6963
         6968#  6968   6970#  6970   6972#  6972   6974#  6974   6976#  6976   6978#  6978   6984#  6984
         6986#  6986   6988#  6988   6990#  6990   6993#  6993   6995#  6995   6997#  6997   6999#  6999
         7001#  7001   7003#  7003   7009#  7009   7011#  7011   7013#  7013   7015#  7015   7018#  7018
         7020#  7020   7022#  7022   7024#  7024   7026#  7026   7028#  7028   7034#  7034   7036#  7036
         7038#  7038   7040#  7040   7043#  7043   7045#  7045   7047#  7047   7049#  7049   7051#  7051
         7053#  7053   7055#  7055   7060#  7060   7062#  7062   7381#  7381   7383#  7383   7385#  7385
         7387#  7387   7389#  7389   7394#  7394   7396#  7396   7398#  7398   7400#  7400   7402#  7402
         7404#  7404   7410#  7410   7412#  7412   7414#  7414   7416#  7416   7419#  7419   7421#  7421
         7423#  7423   7425#  7425   7427#  7427   7429#  7429   7435#  7435   7437#  7437   7439#  7439
         7441#  7441   7444#  7444   7446#  7446   7448#  7448   7450#  7450   7452#  7452   7454#  7454
         7456#  7456   7461#  7461   7463#  7463   7465#  7465   7467#  7467   7469#  7469   7471#  7471
         7473#  7473   7479#  7479   7481#  7481   7483#  7483   7485#  7485   7488#  7488   7490#  7490
         7492#  7492   7494#  7494   7496#  7496   7498#  7498   7500#  7500   7505#  7505   7507#  7507
.CONI      34#   150    252    258    456    462    550
.CONO      34#   359
.DATAI     33#  1031
.DATAO     33#  1030
.LA      2013   2024   2363   3263   3288   3810   3824   3831   3837   3851   3864   3884   4031   4045
         4052   4058   4072   4085   4105   4250   4264   4271   4277   4291   4304   4324   4926   4932
         5558   5829   6112   6369   6378   6635   6838   6974   6976   6999   7001   7009   7024   7026
         7034   7049   7051   7400   7402   7425   7427   7435   7450   7452   7469   7471   7479   7494
         7496
.LADDR   1071   1649   2005   2108   2196   2280   2363   2448   2532   2632   2778   2954   3130   3254
         3263   3266   3275   3288   3370   3446   3526   3596   3680   3804   3898   3900   3902   4025
         4119   4121   4123   4244   4338   4340   4342   4431   4602   4684   4760   4885   4890   4915
         4926   5037   5164   5284   5447   5531   5547   5558   5718   5802   5818   5829   6032   6059
         6066   6080   6091   6112   6291   6318   6325   6339   6351   6378   6555   6582   6589   6603
         6614   6635   6826   6955   7060   7381   7461   7505
.LAND    2005   2363   2954   3254   3256   3258   3260   3804   3824   3851   3864   3884   4025   4045
         4072   4085   4105   4244   4264   4291   4304   4324   4915   5462   5473   5484   5495   5506
         5517   5531   5547   5553   5733   5744   5755   5766   5777   5788   5802   5818   5824   6091
         6351   6614   6974   6976   6984   6999   7001   7024   7034   7049   7400   7402   7410   7425
         7427   7450   7452   7469   7494   7496
.LB      2005   2007   2022   3254   3256   3258   3260   3263   3268   3270   3272   3277   3279   3281
         3283   3804   3806   3810   3819   3824   3846   3851   3879   3884   4025   4027   4031   4040
         4045   4067   4072   4100   4105   4244   4246   4250   4259   4264   4286   4291   4319   4324
         4896   4898   4900   4902   4904   4910   4915   4917   4926   4932   4934   4942   5531   5537
         5547   5551   5802   5808   5818   5822   6072   6080   6082   6091   6097   6105   6331   6339
         6341   6351   6357   6359   6361   6595   6603   6605   6614   6620   6628   6955   6959   6963
         6972   6974   6976   6997   6999   7001   7022   7024   7026   7047   7049   7051   7381   7385
         7389   7398   7400   7402   7423   7425   7427   7448   7450   7452   7467   7469   7471   7492
         7494  7496
.LBAD    3372  3448  3598  4604  5039  5166  5286
.LCCCC   3857  4078  4297
.LCCER   1073  1081  3132  3290  4892  4938  6068  6101  6327  6365  6591  6624
.LCCFZ   3824  3851  3884  4045  4072  4105  4264  4291  4324  6976  7001  7026  7051  7402
         7427  7452  7471  7496
.LCCGC   2009  2018  2108  2196  2280  2365  2448  2532  2632  3815  3833  3842  3866  3875
         4036  4054  4063  4087  4096  4255  4273  4282  4306  4315  4906  6834  6968  6986
         6993  7011  7018  7036  7043  7394  7412  7419  7437  7444  7463  7481  7488
.LCCIA   4919
.LCENA   1073  1081  2009  2018  2108  2196  2280  2365  2448  2532  2632  3132  3290  3815
         3824  3833  3842  3851  3857  3866  3875  3884  4036  4045  4054  4063  4072  4078
         4087  4096  4105  4255  4264  4273  4282  4291  4297  4306  4315  4324  4892  4906
         4919  4938  6068  6101  6327  6365  6591  6624  6834  6968  6976  6986  6993  7001
         7011  7018  7026  7036  7043  7051  7394  7402  7412  7419  7427  7437  7444  7452
         7463  7471  7481  7488  7496
.LCJP    1073  1081  2009  2018  2108  2196  2280  2365  2448  2532  2632  3132  3290  3815
         3824  3833  3842  3851  3857  3866  3875  3884  4036  4045  4054  4063  4072  4078
         4087  4096  4105  4255  4264  4273  4282  4291  4297  4306  4315  4324  4892  4906
         4919  4938  6068  6101  6327  6365  6591  6624  6834  6968  6976  6986  6993  7001
         7011  7018  7026  7036  7043  7051  7394  7402  7412  7419  7427  7437  7444  7452
         7463  7471  7481  7488  7496
.LCJS    4917  5466  5477  5488  5499  5510  5521  5737  5748  5759  5770  5781  5792  6034
         6043  6052  6293  6302  6311  6557  6566  6575
.LCONT   1085  2007  2013  2022  2024  2200  2284  2369  2636  2780  2784  2788  2792  2796
         2800  2804  2808  2812  2816  2820  2824  2828  2832  2836  2840  2844  2848  2852
         2856  2860  2864  2868  2872  2956  2958  2960  3136  3138  3268  3277  3281  3288
         3294  3806  3819  3831  3837  3846  3864  3870  3879  4027  4040  4052  4058  4067
         4085  4091  4100  4246  4259  4271  4277  4286  4304  4310  4319  4896  4932  4934
         4936  4942  5531  5537  5539  5547  5551  5802  5808  5810  5818  5822  6072  6091
         6097  6099  6105  6331  6351  6357  6363  6369  6371  6595  6614  6620  6622  6628
         6832  6838  6972  6974  6984  6990  6997  6999  7009  7015  7022  7024  7034  7040
         7047  7049  7060  7398  7400  7410  7416  7423  7425  7435  7441  7448  7450  7467
         7469  7479  7485  7492  7494  7505
.LCRTN   4944  5553  5824  6084  6343  6607
.LCRY    3268  3272  3277  3281  3806  4027  4246  4917  5464  5475  5486  5497  5508  5519
         5735  5746  5757  5768  5779  5790
.LD      1077  1085  2005  2007  2013  2022  2024  2363  2778  2780  2782  2784  2786  2788
         2790  2792  2794  2796  2798  2800  2802  2804  2806  2808  2810  2812  2814  2816
         2818  2820  2822  2824  2826  2828  2830  2832  2834  2836  2838  2840  2842  2844
         2846  2848  2850  2852  2854  2856  2858  2860  2862  2864  2866  2868  2870  2872
         2954  2956  2958  2960  3136  3138  3140  3254  3256  3258  3260  3263  3266  3268
         3270  3272  3275  3277  3279  3281  3283  3288  3294  3296  3804  3806  3810  3819
         3824  3831  3837  3846  3851  3864  3879  3884  4025  4027  4031  4040  4045  4052
         4058  4067  4072  4085  4100  4105  4244  4246  4250  4259  4264  4271  4277  4286
         4291  4304  4319  4324  4896  4898  4900  4902  4904  4910  4915  4917  4926  4928
         4932  4934  4942  5447  5449  5451  5453  5455  5457  5462  5464  5468  5473  5475
         5479  5484  5486  5490  5495  5497  5501  5506  5508  5512  5517  5519  5523  5531
         5533  5535  5537  5547  5549  5551  5553  5558  5718  5720  5722  5724  5726  5728
         5733  5735  5739  5744  5746  5750  5755  5757  5761  5766  5768  5772  5777  5779
         5783  5788  5790  5794  5802  5804  5806  5808  5818  5820  5822  5824  5829  6034
         6036  6043  6045  6052  6054  6061  6072  6074  6080  6082  6091  6093  6095  6097
         6101  6103  6105  6112  6293  6295  6302  6304  6311  6313  6320  6331  6333  6339
```

```
            6341    6351    6353    6355    6357    6359    6361    6365    6367    6369    6371    6378    6557    6559      SEQ 0676
            6566    6568    6575    6577    6584    6595    6597    6603    6605    6614    6616    6618    6620    6624
            6626    6628    6635    6826    6828    6830    6832    6955    6959    6963    6972    6974    6976    6978
            6984    6997    6999    7001    7003    7009    7022    7024    7026    7028    7034    7047    7049    7051
            7053    7055    7060    7062    7381    7385    7389    7398    7400    7402    7404    7410    7423    7425
            7427    7429    7435    7448    7450    7452    7454    7456    7467    7469    7471    7473    7479    7492
            7494    7496    7498    7500    7505    7507
.LJ         1071    1073    1075    1077    1079    1081    1083    2005    2009    2011    2018    2020    2026    2108
            2110    2112    2196    2198    2280    2282    2286    2363    2365    2367    2371    2448    2450    2452
            2532    2534    2536    2632    2634    2638    2778    2782    2786    2790    2794    2798    2802    2806
            2810    2814    2818    2822    2826    2830    2834    2838    2842    2846    2850    2854    2858    2862
            2866    2870    2954    3130    3132    3134    3140    3254    3256    3258    3260    3263    3266    3270
            3272    3275    3279    3283    3290    3292    3296    3370    3372    3374    3446    3448    3450    3596
            3598    3600    3804    3808    3810    3815    3817    3824    3826    3833    3835    3842    3844    3851
            3857    3859    3866    3868    3875    3877    3884    3886    3891    3898    3900    3902    4025    4029
            4031    4036    4038    4045    4047    4054    4056    4063    4065    4072    4078    4080    4087    4089
            4096    4098    4105    4107    4112    4119    4121    4123    4244    4248    4250    4255    4257    4264
            4266    4273    4275    4282    4284    4291    4297    4299    4306    4308    4315    4317    4324    4326
            4331    4338    4340    4342    4602    4604    4606    4890    4892    4894    4898    4900    4902    4904
            4906    4908    4910    4915    4917    4919    4921    4926    4928    4930    4938    4940    5037    5039
            5041    5164    5166    5168    5284    5286    5288    5447    5449    5451    5453    5455    5457    5462
            5464    5466    5468    5473    5475    5477    5479    5484    5486    5488    5490    5495    5497    5499
            5501    5506    5508    5510    5512    5517    5519    5521    5523    5533    5535    5541    5549    5558
            5560    5718    5720    5722    5724    5726    5728    5733    5735    5737    5739    5744    5746    5748
            5750    5755    5757    5759    5761    5766    5768    5770    5772    5777    5779    5781    5783    5788
            5790    5792    5794    5804    5806    5812    5820    5829    5831    6032    6034    6036    6041    6043
            6045    6050    6052    6054    6059    6061    6066    6068    6070    6074    6080    6082    6093    6095
            6101    6103    6107    6112    6114    6291    6293    6295    6300    6302    6304    6309    6311    6313
            6318    6320    6325    6327    6329    6333    6339    6341    6353    6355    6359    6361    6365    6367
            6373    6378    6380    6555    6557    6559    6564    6566    6568    6573    6575    6577    6582    6584
            6589    6591    6593    6597    6603    6605    6616    6618    6624    6626    6630    6635    6637    6826
            6828    6830    6834    6836    6840    6955    6957    6959    6961    6963    6968    6970    6976    6978
            6986    6988    6993    6995    7001    7003    7011    7013    7018    7020    7026    7028    7036    7038
            7043    7045    7051    7053    7055    7062    7381    7383    7385    7387    7389    7394    7396    7402
            7404    7412    7414    7419    7421    7427    7429    7437    7439    7444    7446    7452    7454    7456
            7461    7463    7465    7471    7473    7481    7483    7488    7490    7496    7498    7500    7507
.LJMAP      1071    1075    1083    2005    2011    2020    2026    2110    2112    2198    2282    2286    2363    2367
            2371    2450    2452    2534    2536    2634    2638    2778    2782    2786    2790    2794    2798    2802
            2806    2810    2814    2818    2822    2826    2830    2834    2838    2842    2846    2850    2854    2858
            2862    2866    2870    2954    3130    3134    3140    3254    3256    3258    3260    3263    3272    3283
            3292    3296    3370    3372    3374    3446    3448    3450    3596    3598    3600    3804    3817    3826
            3835    3844    3859    3868    3877    3886    3891    3898    3900    3902    4025    4038    4047    4056
            4065    4080    4089    4098    4107    4112    4119    4121    4123    4244    4257    4266    4275    4284
            4299    4308    4317    4326    4331    4338    4340    4342    4602    4604    4606    4890    4894    4908
            4910    4915    4921    4926    4940    5037    5039    5041    5164    5166    5168    5284    5286    5288
            5447    5449    5451    5453    5455    5457    5462    5468    5473    5479    5484    5490    5495    5501
            5506    5512    5517    5523    5541    5558    5560    5718    5720    5722    5724    5726    5728    5733
            5739    5744    5750    5755    5761    5766    5772    5777    5783    5788    5794    5812    5829    5831
            6032    6036    6041    6045    6050    6054    6059    6061    6066    6070    6074    6103    6107    6112
            6114    6291    6295    6300    6304    6309    6313    6318    6320    6325    6329    6333    6367    6373
            6378    6380    6555    6559    6564    6568    6573    6577    6582    6584    6589    6593    6597    6626
            6630    6635    6637    6826    6836    6840    6955    6970    6978    6988    6995    7003    7013    7020
            7028    7038    7045    7053    7055    7062    7381    7396    7404    7414    7421    7429    7439    7446
            7454      7456      7461      7465      7473      7483      7490      7498      7500      7507
.LJZ        1087      1649      2202      2373      3526      3680      4431      4684      4760      4885
.LLDCT      1077      3266      3275      3808      4029      4248      4898      4902      4928      5464      5475      5486      5497      5508
            5519      5533      5735      5746      5757      5768      5779      5790      5804      6080      6093      6339      6353      6359
            6603      6616      6828      6957      6961      7383      7387
.LLDLM      2956
.LMGC       1077      1085      2007      2009      2013      2018      2022      2024      2108      2196      2200      2280      2284      2363
            2365      2369      2448      2532      2632      2636      2778      2780      2782      2784      2786      2788      2790      2792
            2794      2796      2798      2800      2802      2804      2806      2808      2810      2812      2814      2816      2818      2820
            2822      2824      2826      2828      2830      2832      2834      2836      2838      2840      2842      2844      2846      2848
            2850      2852      2854      2856      2858      2860      2862      2864      2866      2868      2870      2872      2956      2958
            2960      3136      3138      3288      3294      3526      3680      3815      3819      3831      3833      3837      3842      3846
            3864      3866      3870      3875      3879      4036      4040      4052      4054      4058      4063      4067      4085      4087
            4091      4096      4100      4255      4259      4271      4273      4277      4282      4286      4304      4306      4310      4315
            4319      4896      4898      4902      4906      4910      4915      4934      4936      4942      5447      5449      5451      5453
            5455      5457      5468      5479      5490      5501      5512      5523      5537      5539      5558      5718      5720      5722
            5724      5726      5728      5739      5750      5761      5772      5783      5794      5808      5810      5829      6034      6036
            6043      6045      6052      6054      6061      6072      6097      6099      6105      6112      6293      6295      6302      6304
            6311      6313      6320      6331      6357      6359      6363      6369      6371      6378      6557      6559      6566      6568
            6575      6577      6584      6595      6620      6622      6628      6635      6826      6832      6834      6838      6955      6968
            6972      6978      6984      6986      6990      6993      6997      7003      7009      7011      7015      7018      7022      7028
            7034      7036      7040      7043      7047      7053      7055      7060      7381      7394      7398      7404      7410      7412
            7416      7419      7423      7429      7435      7437      7441      7444      7448      7454      7456      7463      7467      7473
            7479      7481      7485      7488      7492      7498      7500      7505
.LOENA      1085      2007      2024      2363      2780      2784      2788      2792      2796      2800      2804      2808      2812      2816
            2820      2824      2828      2832      2836      2840      2844      2848      2852      2856      2860      2864      2868      2872
            2956      2960      3138      3288      3294      3526      3680      3831      3864      4052      4085      4271      4304      4934
            5537      5558      5808      5829      6097      6112      6357      6369      6378      6620      6635      6832      6984      7009
            7034      7060      7410      7435      7479      7505
.LOR        1077      1085      2013      2022      2024      2778      2780      2782      2784      2786      2788      2790      2792      2794
            2796      2798      2800      2802      2804      2806      2808      2810      2812      2814      2816      2818      2820      2822
            2824      2826      2828      2830      2832      2834      2836      2838      2840      2842      2844      2846      2848      2850
            2852      2854      2856      2858      2860      2862      2864      2866      2868      2870      2872      2956      2958      2960
            3136      3138      3270      3279      3288      3819      3831      3837      3846      3879      4040      4052      4058      4067
            4100      4259      4271      4277      4286      4319      4896      4898      4900      4902      4904      4910      4926      4932
            4934      4942      5447      5449      5451      5453      5455      5457      5468      5479      5490      5501      5512      5523
            5535      5537      5549      5551      5558      5718      5720      5722      5724      5726      5728      5739      5750      5761
            5772      5783      5794      5806      5808      5820      5822      5829      6034      6036      6043      6045      6052      6054
            6061      6072      6080      6082      6095      6097      6105      6112      6293      6295      6302      6304      6311      6313
            6320      6331      6339      6341      6355      6357      6359      6369      6378      6557      6559      6566      6568      6575
            6577      6584      6595      6603      6605      6618      6620      6628      6635      6826      6830      6832      6838      6955
            6959      6963      6972      6978      6997      7003      7009      7022      7028      7047      7053      7055      7060      7381
            7385      7389      7398      7404      7423      7429      7435      7448      7454      7456      7467      7473      7479      7492
            7498      7500      7505
.LPLUS      3268      3272      3277      3281      3806      3810      4027      4031      4246      4250      4917      5464      5475      5486
            5497      5508      5519      5735      5746      5757      5768      5779      5790
.LRDLM      2958      3526      3680
.LRPCT      1079      3270      3279      3810      4031      4250      4900      4904      4930      5535      5549      5806      5820      6082
            6095      6341      6355      6361      6605      6618      6830      6959      6963      7385      7389
.LSOA       1085      2013      2024      2363      2780      2784      2788      2792      2796      2800      2804      2808      2812      2816
            2820      2824      2828      2832      2836      2840      2844      2848      2852      2856      2860      2864      2868      2872
            2956      2960      3138      3254      3256      3258      3260      3263      3288      3831      3837      3864      4052      4058
            4085      4271      4277      4304      4915      4926      5462      5473      5484      5495      5506      5517      5531      5535
```

```
            5547      5549      5553      5558      5733      5744      5755      5766      5777      5788      5802      5806      5818      5820
```

```
665
        5547   5549   5553   5558   5733   5744   5755   5766   5777   5788   5802   5806   5818   5820     SEQ 0678
        5824   5829   6080   6091   6095   6112   6339   6351   6355   6369   6378   6603   6614   6618
        6635   6830   6832   6838   6984   7009   7034   7060   7410   7435   7479   7505
.LSOB   2007   3268   3270   3272   3277   3279   3281   3283   3806   4027   4246   4900   4904   4917
        4934   6082   6341   6361   6605   6959   6963   7385   7389
.LSAB   3810   3824   3851   3884   4031   4045   4072   4105   4250   4264   4291   4324   4932   5464
        5475   5486   5497   5508   5519   5537   5551   5735   5746   5757   5768   5779   5790   5808
        5822   6097   6357   6620   6974   6976   6999   7001   7024   7026   7049   7051   7400   7402
        7425   7427   7450   7452   7469   7471   7494   7496
.LSDO   1077   2005   2022   2778   2782   2786   2790   2794   2798   2802   2806   2810   2814   2818
        2822   2826   2830   2834   2838   2842   2846   2850   2854   2858   2862   2866   2870   2954
        2958   3136   3804   3819   3846   3879   4025   4040   4067   4100   4244   4259   4286   4319
        4896   4898   4902   4910   4942   5447   5449   5451   5453   5455   5457   5468   5479   5490
        5501   5512   5523   5718   5720   5722   5724   5726   5728   5739   5750   5761   5772   5783
        5794   6034   6036   6043   6045   6052   6054   6061   6072   6105   6293   6295   6302   6304
        6311   6313   6320   6331   6359   6557   6559   6566   6568   6575   6577   6584   6595   6628
        6826   6955   6972   6978   6997   7003   7022   7028   7047   7053   7055   7381   7398   7404
        7423   7429   7448   7454   7456   7467   7473   7492   7498   7500
.LSELE  1077   1085   2007   2009   2013   2018   2022   2024   2108   2196   2200   2280   2284   2363
        2365   2369   2448   2532   2632   2636   2780   2784   2788   2792   2796   2800   2804   2808
        2812   2816   2820   2824   2828   2832   2836   2840   2844   2848   2852   2856   2860   2864
        2868   2872   2960   3136   3138   3288   3294   3526   3680   3815   3819   3831   3833   3837
        3842   3846   3864   3866   3870   3875   3879   4036   4040   4052   4054   4058   4063   4067
        4085   4087   4091   4096   4100   4255   4259   4271   4273   4277   4282   4286   4304   4306     MWORD
        4310   4315   4319   4896   4906   4910   4915   4934   4936   4942   5537   5539   5558   5808
        5810   5829   6072   6097   6099   6105   6112   6331   6357   6363   6369   6371   6378   6595
        6620   6622   6628   6635   6832   6834   6838   6968   6972   6984   6986   6990   6993   6997
        7009   7011   7015   7018   7022   7034   7036   7040   7043   7047   7060   7394   7398   7410
        7412   7416   7419   7423   7435   7437   7441   7444   7448   7463   7467   7479   7481   7485
        7488   7492   7505
.LSKCN  2778   2782   2786   2790   2794   2798   2802   2806   2810   2814   2818   2822   2826   2830
        2834   2838   2842   2846   2850   2854   2858   2862   2866   2870   4898   4902   5447   5449
        5451   5453   5455   5457   5468   5479   5490   5501   5512   5523   5718   5720   5722   5724
        5726   5728   5739   5750   5761   5772   5783   5794   6034   6036   6043   6045   6052   6054
        6061   6293   6295   6302   6304   6311   6313   6320   6359   6557   6559   6566   6568   6575
        6577   6584   6826   6955   6978   7003   7028   7053   7055   7381   7404   7429   7454   7456
        7473   7498   7500
.LXNOR  2007   3263   3283
.LXOR   7026   7051   7471
.MA     2013   2024   2363   3263   3288   3810   3824   3831   3837   3851   3864   3884   4031   4045
        4052   4058   4072   4085   4105   4250   4264   4271   4277   4291   4304   4324   4926   4932
        5558   5829   6112   6369   6378   6635   6838   6974   6976   6999   7001   7009   7024   7026
        7034   7049   7051   7400   7402   7425   7427   7435   7450   7452   7469   7471   7479   7494
        7496
.MAND   2005   2363   2954   3254   3256   3258   3260   3804   3824   3851   3864   3884   4025   4045
        4072   4085   4105   4244   4264   4291   4304   4324   4915   5462   5473   5484   5495   5506
        5517   5531   5547   5553   5733   5744   5755   5766   5777   5788   5802   5818   5824   6091
        6351   6614   6974   6976   6984   6999   7001   7024   7034   7049   7400   7402   7410   7425
        7427   7450   7452   7469   7494   7496
.MB     2005   2007   2022   3254   3256   3258   3260   3263   3268   3270   3272   3277   3279   3281
        3283   3804   3806   3810   3819   3824   3846   3851   3879   3884   4025   4027   4031   4040
        4045   4067   4072   4100   4105   4244   4246   4250   4259   4264   4286   4291   4319   4324
        4896   4898   4900   4902   4904   4910   4915   4917   4926   4932   4934   4942   5531   5537

666     5547   5551   5802   5808   5818   5822   6072   6080   6082   6091   6097   6105   6331   6339     SEQ 0679
```

```
        5547  5551  5802  5808  5818  5822  6072  6080  6082  6091  6097  6105  6331  6339     SEQ 0679
        6341  6351  6357  6359  6361  6595  6603  6605  6614  6620  6628  6955  6959  6963
        6972  6974  6976  6997  6999  7001  7022  7024  7026  7047  7049  7051  7381  7385
        7389  7398  7400  7402  7423  7425  7427  7448  7450  7452  7467  7469  7471  7492
        7494  7496
.MBAD   3372  3448  3598  4604  5039  5166  5286
.MCCCC  3857  4078  4297
.MCCFR  1073  1081  3132  3290  4892  4938  6068  6101  6327  6365  6591  6624
.MCCFZ  3824  3851  3884  4045  4072  4105  4264  4291  4324  6976  7001  7026  7051  7402
        7427  7452  7471  7496
.MCCGC  2009  2018  2108  2196  2280  2365  2448  2532  2632  3815  3833  3842  3866  3875
        4036  4054  4063  4087  4096  4255  4273  4282  4306  4315  4906  6834  6968  6986
        6993  7011  7018  7036  7043  7394  7412  7419  7437  7444  7463  7481  7488
.MCCIA  4919
.MCENA  1073  1081  2009  2018  2108  2196  2280  2365  2448  2532  2632  3132  3290  3815
        3824  3833  3842  3851  3857  3866  3875  3884  4036  4045  4054  4063  4072  4078
        4087  4096  4105  4255  4264  4273  4282  4291  4297  4306  4315  4324  4892  4906
        4919  4938  6068  6101  6327  6365  6591  6624  6834  6968  6976  6986  6993  7001
        7011  7018  7026  7036  7043  7051  7394  7402  7412  7419  7427  7437  7444  7452
        7463  7471  7481  7488  7496
.MCJP   1073  1081  2009  2018  2108  2196  2280  2365  2448  2532  2632  3132  3290  3815
        3824  3833  3842  3851  3857  3866  3875  3884  4036  4045  4054  4063  4072  4078
        4087  4096  4105  4255  4264  4273  4282  4291  4297  4306  4315  4324  4892  4906
        4919  4938  6068  6101  6327  6365  6591  6624  6834  6968  6976  6986  6993  7001
        7011  7018  7026  7036  7043  7051  7394  7402  7412  7419  7427  7437  7444  7452
        7463  7471  7481  7488  7496
.MCJS   4917  5466  5477  5488  5499  5510  5521  5737  5748  5759  5770  5781  5792  6034
        6043  6052  6293  6302  6311  6557  6566  6575
.MCONT  1085  2007  2013  2022  2024  2200  2284  2369  2636  2780  2784  2788  2792  2796
        2800  2804  2808  2812  2816  2820  2824  2828  2832  2836  2840  2844  2848  2852
        2856  2860  2864  2868  2872  2956  2958  2960  3136  3138  3268  3277  3281  3288
        3294  3806  3819  3831  3837  3846  3864  3870  3879  4027  4040  4052  4058  4067
        4085  4091  4100  4246  4259  4271  4277  4286  4304  4310  4319  4896  4932  4934
        4936  4942  5531  5537  5539  5547  5551  5802  5808  5810  5818  5822  6072  6091
        6097  6099  6105  6331  6351  6357  6363  6369  6371  6595  6614  6620  6622  6628
        6832  6838  6972  6974  6984  6990  6997  6999  7009  7015  7022  7024  7034  7040
        7047  7049  7060  7398  7400  7410  7416  7423  7425  7435  7441  7448  7450  7467
        7469  7479  7485  7492  7494  7505
.MCRTN  4944  5553  5824  6084  6343  6607
.MCRY   3268  3272  3277  3281  3806  4027  4246  4917  5464  5475  5486  5497  5508  5519
        5735  5746  5757  5768  5779  5790
.MD     1077  1085  2005  2007  2013  2022  2024  2363  2778  2780  2782  2784  2786  2788
        2790  2792  2794  2796  2798  2800  2802  2804  2806  2808  2810  2812  2814  2816
        2818  2820  2822  2824  2826  2828  2830  2832  2834  2836  2838  2840  2842  2844
        2846  2848  2850  2852  2854  2856  2858  2860  2862  2864  2866  2868  2870  2872
        2954  2956  2958  2960  3136  3138  3140  3254  3256  3258  3260  3263  3266  3268
        3270  3272  3275  3277  3279  3281  3283  3288  3294  3296  3804  3806  3810  3819
        3824  3831  3837  3846  3851  3864  3879  3884  4025  4027  4031  4040  4045  4052
        4058  4067  4072  4085  4100  4105  4244  4246  4250  4259  4264  4271  4277  4286
        4291  4304  4319  4324  4896  4898  4900  4902  4904  4910  4915  4917  4926  4928
        4932  4934  4942  5447  5449  5451  5453  5455  5457  5462  5464  5468  5473  5475
        5479  5484  5486  5490  5495  5497  5501  5506  5508  5512  5517  5519  5523  5531
        5533  5535  5537  5547  5549  5551  5553  5558  5718  5720  5722  5724  5726  5728
```

```
          5733  5735  5739  5744  5746  5750  5755  5757  5761  5766  5768  5772  5777  5779        SEQ 0680
          5783  5788  5790  5794  5802  5804  5806  5808  5818  5820  5822  5824  5829  6034
          6036  6043  6045  6052  6054  6061  6072  6074  6080  6082  6091  6093  6095  6097
          6101  6103  6105  6112  6293  6295  6302  6304  6311  6313  6320  6331  6333  6339
          6341  6351  6353  6355  6357  6359  6361  6365  6367  6369  6371  6378  6557  6559
          6566  6568  6575  6577  6584  6595  6597  6603  6605  6614  6616  6618  6620  6624
          6626  6628  6635  6826  6828  6830  6832  6955  6959  6963  6972  6974  6976  6978
          6984  6997  6999  7001  7003  7009  7022  7024  7026  7028  7034  7047  7049  7051
          7053  7055  7060  7062  7381  7385  7389  7398  7400  7402  7404  7410  7423  7425
          7427  7429  7435  7448  7450  7452  7454  7456  7467  7469  7471  7473  7479  7492
          7494  7496  7498  7500  7505  7507
.MJ       1071  1073  1075  1077  1079  1081  1083  2005  2009  2011  2018  2020  2026  2108
          2110  2112  2196  2198  2280  2282  2286  2363  2365  2367  2371  2448  2450  2452
          2532  2534  2536  2632  2634  2638  2778  2782  2786  2790  2794  2798  2802  2806
          2810  2814  2818  2822  2826  2830  2834  2838  2842  2846  2850  2854  2858  2862
          2866  2870  2954  3130  3132  3134  3140  3254  3256  3258  3260  3263  3266  3270
          3272  3275  3279  3283  3290  3292  3296  3370  3372  3374  3446  3448  3450  3596
          3598  3600  3804  3808  3810  3815  3817  3824  3826  3833  3835  3842  3844  3851
          3857  3859  3866  3868  3875  3877  3884  3886  3891  3898  3900  3902  4025  4029
          4031  4036  4038  4045  4047  4054  4056  4063  4065  4072  4078  4080  4087  4089
          4096  4098  4105  4107  4112  4119  4121  4123  4244  4248  4250  4255  4257  4264
          4266  4273  4275  4282  4284  4291  4297  4299  4306  4308  4315  4317  4324  4326
          4331  4338  4340  4342  4602  4604  4606  4890  4892  4894  4898  4900  4902  4904
          4906  4908  4910  4915  4917  4919  4921  4926  4928  4930  4938  4940  5037  5039
          5041  5164  5166  5168  5284  5286  5288  5447  5449  5451  5453  5455  5457  5462
          5464  5466  5468  5473  5475  5477  5479  5484  5486  5488  5490  5495  5497  5499
          5501  5506  5508  5510  5512  5517  5519  5521  5523  5533  5535  5541  5549  5558
          5560  5718  5720  5722  5724  5726  5728  5733  5735  5737  5739  5744  5746  5748
          5750  5755  5757  5759  5761  5766  5768  5770  5772  5777  5779  5781  5783  5788
          5790  5792  5794  5804  5806  5812  5820  5829  5831  6032  6034  6036  6041  6043
          6045  6050  6052  6054  6059  6061  6066  6068  6070  6074  6080  6082  6093  6095
          6101  6103  6107  6112  6114  6291  6293  6295  6300  6302  6304  6309  6311  6313
          6318  6320  6325  6327  6329  6333  6339  6341  6353  6355  6359  6361  6365  6367
          6373  6378  6380  6555  6557  6559  6564  6566  6568  6573  6575  6577  6582  6584
          6589  6591  6593  6597  6603  6605  6616  6618  6624  6626  6630  6635  6637  6826
          6828  6830  6834  6836  6840  6955  6957  6959  6961  6963  6968  6970  6976  6978
          6986  6988  6993  6995  7001  7003  7011  7013  7018  7020  7026  7028  7036  7038
          7043  7045  7051  7053  7055  7062  7381  7383  7385  7387  7389  7394  7396  7402
          7404  7412  7414  7419  7421  7427  7429  7437  7439  7444  7446  7452  7454  7456
          7461  7463  7465  7471  7473  7481  7483  7488  7490  7496  7498  7500  7507
.MJMAP    1071  1075  1083  2005  2011  2020  2026  2110  2112  2198  2282  2286  2363  2367
          2371  2450  2452  2534  2536  2634  2638  2778  2782  2786  2790  2794  2798  2802
          2806  2810  2814  2818  2822  2826  2830  2834  2838  2842  2846  2850  2854  2858
          2862  2866  2870  2954  3130  3134  3140  3254  3256  3258  3260  3263  3272  3283
          3292  3296  3370  3372  3374  3446  3448  3450  3596  3598  3600  3804  3817  3826
          3835  3844  3859  3868  3877  3886  3891  3898  3900  3902  4025  4038  4047  4056
          4065  4080  4089  4098  4107  4112  4119  4121  4123  4244  4257  4266  4275  4284
          4299  4308  4317  4326  4331  4338  4340  4342  4602  4604  4606  4890  4894  4908
          4910  4915  4921  4926  4940  5037  5039  5041  5164  5166  5168  5284  5286  5288
          5447  5449  5451  5453  5455  5457  5462  5468  5473  5479  5484  5490  5495  5501
          5506  5512  5517  5523  5541  5558  5560  5718  5720  5722  5724  5726  5728  5733
          5739  5744  5750  5755  5761  5766  5772  5777  5783  5788  5794  5812  5829  5831
          6032  6036  6041  6045  6050  6054  6059  6061  6066  6070  6074  6103  6107  6112
```

```
          6114  6291  6295  6300  6304  6309  6313  6318  6320  6325  6329  6333  6367  6373
```

```
668
```

```
          6114    6291    6295    6300    6304    6309    6313    6318    6320    6325    6329    6333    6367    6373       SEQ 0681
          6378    6380    6555    6559    6564    6568    6573    6577    6582    6584    6589    6593    6597    6626
          6630    6635    6637    6826    6836    6840    6955    6970    6978    6988    6995    7003    7013    7020
          7028    7038    7045    7053    7055    7062    7381    7396    7404    7414    7421    7429    7439    7446
          7454    7456    7461    7465    7473    7483    7490    7498    7500    7507
.MJZ      1087    1649    2202    2373    3526    3680    4431    4684    4760    4885
.MLDCT    1077    3266    3275    3808    4029    4248    4898    4902    4928    5464    5475    5486    5497    5508
          5519    5533    5735    5746    5757    5768    5779    5790    5804    6080    6093    6339    6353    6359
          6603    6616    6828    6957    6961    7383    7387
.MLDLM    2956
.MMGC     1077    1085    2007    2009    2013    2018    2022    2024    2108    2196    2200    2280    2284    2363
          2365    2369    2448    2532    2632    2636    2778    2780    2782    2784    2786    2788    2790    2792
          2794    2796    2798    2800    2802    2804    2806    2808    2810    2812    2814    2816    2818    2820
          2822    2824    2826    2828    2830    2832    2834    2836    2838    2840    2842    2844    2846    2848
          2850    2852    2854    2856    2858    2860    2862    2864    2866    2868    2870    2872    2956    2958
          2960    3136    3138    3288    3294    3526    3680    3815    3819    3831    3833    3837    3842    3846
          3864    3866    3870    3875    3879    4036    4040    4052    4054    4058    4063    4067    4085    4087
          4091    4096    4100    5255    4259    4271    4273    4277    4282    4286    4304    4306    4310    4315
          4319    4896    4898    4902    4906    4910    4915    4934    4936    4942    5447    5449    5451    5453
          5455    5457    5468    5479    5490    5501    5512    5523    5537    5539    5558    5718    5720    5722
          5724    5726    5728    5739    5750    5761    5772    5783    5794    5808    5810    5829    6034    6036
          6043    6045    6052    6054    6061    6072    6097    6099    6105    6112    6293    6295    6302    6304
          6311    6313    6320    6331    6357    6359    6363    6369    6371    6378    6557    6559    6566    6568
          6575    6577    6584    6595    6620    6622    6628    6635    6826    6832    6834    6838    6955    6968
          6972    6978    6984    6986    6990    6993    6997    7003    7009    7011    7015    7018    7022    7028
          7034    7036    7040    7043    7047    7053    7055    7060    7381    7394    7398    7404    7410    7412
          7416    7419    7423    7429    7435    7437    7441    7444    7448    7454    7456    7463    7467    7473
          7479    7481    7485    7488    7492    7498    7500    7505
.MOENA    1085    2007    2024    2363    2780    2784    2788    2792    2796    2800    2804    2808    2812    2816
          2820    2824    2828    2832    2836    2840    2844    2848    2852    2856    2860    2864    2868    2872
          2956    2960    3138    3288    3294    3526    3680    3831    3864    4052    4085    4271    4304    4934
          5537    5558    5808    5829    6097    6112    6357    6369    6378    6620    6635    6832    6984    7009
          7034    7060    7410    7435    7479    7505
.MOR      1077    1085    2013    2022    2024    2778    2780    2782    2784    2786    2788    2790    2792    2794
          2796    2798    2800    2802    2804    2806    2808    2810    2812    2814    2816    2818    2820    2822
          2824    2826    2828    2830    2832    2834    2836    2838    2840    2842    2844    2846    2848    2850
          2852    2854    2856    2858    2860    2862    2864    2866    2868    2870    2872    2956    2958    2960
          3136    3138    3270    3279    3288    3819    3831    3837    3846    3879    4040    4052    4058    4067
          4100    4259    4271    4277    4286    4319    4896    4898    4900    4902    4904    4910    4926    4932
          4934    4942    5447    5449    5451    5453    5455    5457    5468    5479    5490    5501    5512    5523
          5535    5537    5549    5551    5558    5718    5720    5722    5724    5726    5728    5739    5750    5761
          5772    5783    5794    5806    5808    5820    5822    5829    6034    6036    6043    6045    6052    6054
          6061    6072    6080    6082    6095    6097    6105    6112    6293    6295    6302    6304    6311    6313
          6320    6331    6339    6341    6355    6357    6359    6369    6378    6557    6559    6566    6568    6575
          6577    6584    6595    6603    6605    6618    6620    6628    6635    6826    6830    6832    6838    6955
          6959    6963    6972    6978    6997    7003    7009    7022    7028    7047    7053    7055    7060    7381
          7385    7389    7398    7404    7423    7429    7435    7448    7454    7456    7467    7473    7479    7492
          7498    7500    7505
.MPLUS    3268    3272    3277    3281    3806    3810    4027    4031    4246    4250    4917    5464    5475    5486
          5497    5508    5519    5735    5746    5757    5768    5779    5790
.MRDLM    2958    3526    3680
.MRPCT    1079    3270    3279    3810    4031    4250    4900    4904    4930    5535    5549    5806    5820    6082
          6095    6341    6355    6361    6605    6618    6830    6959    6963    7385    7389
```

```
DFPTA
DFPTA
```

```
MSOA      1085    2013    2024    2363    2780    2784    2788    2792    2796    2800    2804    2808    2812    2816
```

```
.MSOA     1085   2013   2024   2363   2780   2784   2788   2792   2796   2800   2804   2808   2812   2816
          2820   2824   2828   2832   2836   2840   2844   2848   2852   2856   2860   2864   2868   2872      SEQ 0682
          2956   2960   3138   3254   3256   3258   3260   3263   3288   3831   3837   3864   4052   4058
          4085   4271   4277   4304   4915   4926   5462   5473   5484   5495   5506   5517   5531   5535
          5547   5549   5553   5558   5733   5744   5755   5766   5777   5788   5802   5806   5818   5820
          5824   5829   6080   6091   6095   6112   6339   6351   6355   6369   6378   6603   6614   6618
          6635   6830   6832   6838   6984   7009   7034   7060   7410   7435   7479   7505
.MSOB     2007   3268   3270   3272   3277   3279   3281   3283   3806   4027   4246   4900   4904   4917
          4934   6082   6341   6361   6605   6959   6963   7385   7389
.MSAB     3810   3824   3851   3884   4031   4045   4072   4105   4250   4264   4291   4324   4932   5464
          5475   5486   5497   5508   5519   5537   5551   5735   5746   5757   5768   5779   5790   5808
          5822   6097   6357   6620   6974   6976   6999   7001   7024   7026   7049   7051   7400   7402
          7425   7427   7450   7452   7469   7471   7494   7496
.MSDO     1077   2005   2022   2778   2782   2786   2790   2794   2798   2802   2806   2810   2814   2818
          2822   2826   2830   2834   2838   2842   2846   2850   2854   2858   2862   2866   2870   2954
          2958   3136   3804   3819   3846   3879   4025   4040   4067   4100   4244   4259   4286   4319
          4896   4898   4902   4910   4942   5447   5449   5451   5453   5455   5457   5468   5479   5490
          5501   5512   5523   5718   5720   5722   5724   5726   5728   5739   5750   5761   5772   5783
          5794   6034   6036   6043   6045   6052   6054   6061   6072   6105   6293   6295   6302   6304
          6311   6313   6320   6331   6359   6557   6559   6566   6568   6575   6577   6584   6595   6628
          6826   6955   6972   6978   6997   7003   7022   7028   7047   7053   7055   7381   7398   7404
          7423   7429   7448   7454   7456   7467   7473   7492   7498   7500
.MSELE    1077   1085   2007   2009   2013   2018   2022   2024   2108   2196   2200   2280   2284   2363
          2365   2369   2448   2532   2632   2636   2780   2784   2788   2792   2796   2800   2804   2808
          2812   2816   2820   2824   2828   2832   2836   2840   2844   2848   2852   2856   2860   2864
          2868   2872   2960   3136   3138   3288   3294   3526   3680   3815   3819   3831   3833   3837
          3842   3846   3864   3866   3870   3875   3879   4036   4040   4052   4054   4058   4063   4067
          4085   4087   4091   4096   4100   4255   4259   4271   4273   4277   4282   4286   4304   4306
          4310   4315   4319   4896   4906   4910   4915   4934   4936   4942   5537   5539   5558   5808
          5810   5829   6072   6097   6099   6105   6112   6331   6357   6363   6369   6371   6378   6595
          6620   6622   6628   6635   6832   6834   6838   6968   6972   6984   6986   6990   6993   6997
          7009   7011   7015   7018   7022   7034   7036   7040   7043   7047   7060   7394   7398   7410
          7412   7416   7419   7423   7435   7437   7441   7444   7448   7463   7467   7479   7481   7485
          7488   7492   7505
.MSKCN    2778   2782   2786   2790   2794   2798   2802   2806   2810   2814   2818   2822   2826   2830
          2834   2838   2842   2846   2850   2854   2858   2862   2866   2870   4898   4902   5447   5449
          5451   5453   5455   5457   5468   5479   5490   5501   5512   5523   5718   5720   5722   5724
          5726   5728   5739   5750   5761   5772   5783   5794   6034   6036   6043   6045   6052   6054
          6061   6293   6295   6302   6304   6311   6313   6320   6359   6557   6559   6566   6568   6575
          6577   6584   6826   6955   6978   7003   7028   7053   7055   7381   7404   7429   7454   7456
          7473   7498   7500
.MXNOR    2007   3263   3283
.MXOR     7026   7051   7471
.PION      34#   4998   5106   5223
.RA       2013   2024   2363   3263   3288   3810   3824   3831   3837   3851   3864   3884   4031   4045
          4052   4058   4072   4085   4105   4250   4264   4271   4277   4291   4304   4324   4926   4932
          5558   5829   6112   6369   6378   6635   6838   6974   6976   6999   7001   7009   7024   7026
          7034   7049   7051   7400   7402   7425   7427   7435   7450   7452   7469   7471   7479   7494
          7496
.RAND     2005   2363   2954   3254   3256   3258   3260   3804   3824   3851   3864   3884   4025   4045
          4072   4085   4105   4244   4264   4291   4304   4324   4915   5462   5473   5484   5495   5506
          5517   5531   5547   5553   5733   5744   5755   5766   5777   5788   5802   5818   5824   6091
          6351   6614   6974   6976   6984   6999   7001   7024   7034   7049   7400   7402   7410   7425
          7427  7450  7452  7469  7494  7496
.RB       2005  2007  2022  3254  3256  3258  3260  3263  3268  3270  3272  3277  3279  3281
          3283  3804  3806  3810  3819  3824  3846  3851  3879  3884  4025  4027  4031  4040
          4045  4067  4072  4100  4105  4244  4246  4250  4259  4264  4286  4291  4319  4324
          4896  4898  4900  4902  4904  4910  4915  4917  4926  4932  4934  4942  5531  5537
          5547  5551  5802  5808  5818  5822  6072  6080  6082  6091  6097  6105  6331  6339
          6341  6351  6357  6359  6361  6595  6603  6605  6614  6620  6628  6955  6959  6963
          6972  6974  6976  6997  6999  7001  7022  7024  7026  7047  7049  7051  7381  7385
          7389  7398  7400  7402  7423  7425  7427  7448  7450  7452  7467  7469  7471  7492
          7494  7496
.RBAD     3372  3448  3598  4604  5039  5166  5286
.RCCCC    3857  4078  4297
.RCCER    1073  1081  3132  3290  4892  4938  6068  6101  6327  6365  6591  6624
.RCCFZ    3824  3851  3884  4045  4072  4105  4264  4291  4324  6976  7001  7026  7051  7402
          7427  7452  7471  7496
.RCCGC    2009  2018  2108  2196  2280  2365  2448  2532  2632  3815  3833  3842  3866  3875
          4036  4054  4063  4087  4096  4255  4273  4282  4306  4315  4906  6834  6968  6986
          6993  7011  7018  7036  7043  7394  7412  7419  7437  7444  7463  7481  7488
.RCCIA    4919
.RCENA    1073  1081  2009  2018  2108  2196  2280  2365  2448  2532  2632  3132  3290  3815
          3824  3833  3842  3851  3857  3866  3875  3884  4036  4045  4054  4063  4072  4078
          4087  4096  4105  4255  4264  4273  4282  4291  4297  4306  4315  4324  4892  4906
          4919  4938  6068  6101  6327  6365  6591  6624  6834  6968  6976  6986  6993  7001
          7011  7018  7026  7036  7043  7051  7394  7402  7412  7419  7427  7437  7444  7452
          7463  7471  7481  7488  7496
.RCJP     1073  1081  2009  2018  2108  2196  2280  2365  2448  2532  2632  3132  3290  3815
          3824  3833  3842  3851  3857  3866  3875  3884  4036  4045  4054  4063  4072  4078
          4087  4096  4105  4255  4264  4273  4282  4291  4297  4306  4315  4324  4892  4906
          4919  4938  6068  6101  6327  6365  6591  6624  6834  6968  6976  6986  6993  7001
          7011  7018  7026  7036  7043  7051  7394  7402  7412  7419  7427  7437  7444  7452
          7463  7471  7481  7488  7496
.RCJS     4917  5466  5477  5488  5499  5510  5521  5737  5748  5759  5770  5781  5792  6034
          6043  6052  6293  6302  6311  6557  6566  6575
.RCONT    1085  2007  2013  2022  2024  2200  2284  2369  2636  2780  2784  2788  2792  2796
          2800  2804  2808  2812  2816  2820  2824  2828  2832  2836  2840  2844  2848  2852
          2856  2860  2864  2868  2872  2956  2958  2960  3136  3138  3268  3277  3281  3288
          3294  3806  3819  3831  3837  3846  3864  3870  3879  4027  4040  4052  4058  4067
          4085  4091  4100  4246  4259  4271  4277  4286  4304  4310  4319  4896  4932  4934
          4936  4942  5531  5537  5539  5547  5551  5802  5808  5810  5818  5822  6072  6091
          6097  6099  6105  6331  6351  6357  6363  6369  6371  6595  6614  6620  6622  6628
          6832  6838  6972  6974  6984  6990  6997  6999  7009  7015  7022  7024  7034  7040
          7047  7049  7060  7398  7400  7410  7416  7423  7425  7435  7441  7448  7450  7467
          7469  7479  7485  7492  7494  7505
.RCRTN    4944  5553  5824  6084  6343  6607
.RCRY     3268  3272  3277  3281  3806  4027  4246  4917  5464  5475  5486  5497  5508  5519
          5735  5746  5757  5768  5779  5790
.RD       1077  1085  2005  2007  2013  2022  2024  2363  2778  2780  2782  2784  2786  2788
          2790  2792  2794  2796  2798  2800  2802  2804  2806  2808  2810  2812  2814  2816
          2818  2820  2822  2824  2826  2828  2830  2832  2834  2836  2838  2840  2842  2844
          2846  2848  2850  2852  2854  2856  2858  2860  2862  2864  2866  2868  2870  2872
          2954  2956  2958  2960  3136  3138  3140  3254  3256  3258  3260  3263  3266  3268
          3270  3272  3275  3277  3279  3281  3283  3288  3294  3296  3804  3806  3810  3819
          3824  3831  3837  3846  3851  3864  3879  3884  4025  4027  4031  4040  4045  4052
          4058  4067  4072  4085  4100  4105  4244  4246  4250  4259  4264  4271  4277  4286    SEQ 0684
          4291  4304  4319  4324  4896  4898  4900  4902  4904  4910  4915  4917  4926  4928
          4932  4934  4942  5447  5449  5451  5453  5455  5457  5462  5464  5468  5473  5475
          5479  5484  5486  5490  5495  5497  5501  5506  5508  5512  5517  5519  5523  5531
          5533  5535  5537  5547  5549  5551  5553  5558  5718  5720  5722  5724  5726  5728
          5733  5735  5739  5744  5746  5750  5755  5757  5761  5766  5768  5772  5777  5779
          5783  5788  5790  5794  5802  5804  5806  5808  5818  5820  5822  5824  5829  6034
          6036  6043  6045  6052  6054  6061  6072  6074  6080  6082  6091  6093  6095  6097
          6101  6103  6105  6112  6293  6295  6302  6304  6311  6313  6320  6331  6333  6339
          6341  6351  6353  6355  6357  6359  6361  6365  6367  6369  6371  6378  6557  6559
          6566  6568  6575  6577  6584  6595  6597  6603  6605  6614  6616  6618  6620  6624
          6626  6628  6635  6826  6828  6830  6832  6955  6959  6963  6972  6974  6976  6978
          6984  6997  6999  7001  7003  7009  7022  7024  7026  7028  7034  7047  7049  7051
          7053  7055  7060  7062  7381  7385  7389  7398  7400  7402  7404  7410  7423  7425
          7427  7429  7435  7448  7450  7452  7454  7456  7467  7469  7471  7473  7479  7492
          7494  7496  7498  7500  7505  7507
.RJ       1071  1073  1075  1077  1079  1081  1083  2005  2009  2011  2018  2020  2026  2108
          2110  2112  2196  2198  2280  2282  2286  2363  2365  2367  2371  2448  2450  2452
          2532  2534  2536  2632  2634  2638  2778  2782  2786  2790  2794  2798  2802  2806
          2810  2814  2818  2822  2826  2830  2834  2838  2842  2846  2850  2854  2858  2862
          2866  2870  2954  3130  3132  3134  3140  3254  3256  3258  3260  3263  3266  3270
          3272  3275  3279  3283  3290  3292  3296  3370  3372  3374  3446  3448  3450  3596
          3598  3600  3804  3808  3810  3815  3817  3824  3826  3833  3835  3842  3844  3851
          3857  3859  3866  3868  3875  3877  3884  3886  3891  3898  3900  3902  4025  4029
          4031  4036  4038  4045  4047  4054  4056  4063  4065  4072  4078  4080  4087  4089
          4096  4098  4105  4107  4112  4119  4121  4123  4244  4248  4250  4255  4257  4264
          4266  4273  4275  4282  4284  4291  4297  4299  4306  4308  4315  4317  4324  4326
          4331  4338  4340  4342  4602  4604  4606  4890  4892  4894  4898  4900  4902  4904
          4906  4908  4910  4915  4917  4919  4921  4926  4928  4930  4938  4940  5037  5039
          5041  5164  5166  5168  5284  5286  5288  5447  5449  5451  5453  5455  5457  5462
          5464  5466  5468  5473  5475  5477  5479  5484  5486  5488  5490  5495  5497  5499
          5501  5506  5508  5510  5512  5517  5519  5521  5523  5533  5535  5541  5549  5558
          5560  5718  5720  5722  5724  5726  5728  5733  5735  5737  5739  5744  5746  5748
          5750  5755  5757  5759  5761  5766  5768  5770  5772  5777  5779  5781  5783  5788
          5790  5792  5794  5804  5806  5812  5820  5829  5831  6032  6034  6036  6041  6043
          6045  6050  6052  6054  6059  6061  6066  6068  6070  6074  6080  6082  6093  6095
          6101  6103  6107  6112  6114  6291  6293  6295  6300  6302  6304  6309  6311  6313
          6318  6320  6325  6327  6329  6333  6339  6341  6353  6355  6359  6361  6365  6367
          6373  6378  6380  6555  6557  6559  6564  6566  6568  6573  6575  6577  6582  6584
          6589  6591  6593  6597  6603  6605  6616  6618  6624  6626  6630  6635  6637  6826
          6828  6830  6834  6836  6840  6955  6957  6959  6961  6963  6968  6970  6976  6978
          6986  6988  6993  6995  7001  7003  7011  7013  7018  7020  7026  7028  7036  7038
          7043  7045  7051  7053  7055  7062  7381  7383  7385  7387  7389  7394  7396  7402
          7404  7412  7414  7419  7421  7427  7429  7437  7439  7444  7446  7452  7454  7456
          7461  7463  7465  7471  7473  7481  7483  7488  7490  7496  7498  7500  7507
.RJMAP    1071  1075  1083  2005  2011  2020  2026  2110  2112  2198  2282  2286  2363  2367
          2371  2450  2452  2534  2536  2634  2638  2778  2782  2786  2790  2794  2798  2802
          2806  2810  2814  2818  2822  2826  2830  2834  2838  2842  2846  2850  2854  2858
          2862  2866  2870  2954  3130  3134  3140  3254  3256  3258  3260  3263  3272  3283
          3292  3296  3370  3372  3374  3446  3448  3450  3596  3598  3600  3804  3817  3826
          3835  3844  3859  3868  3877  3886  3891  3898  3900  3902  4025  4038  4047  4056
          4065  4080  4089  4098  4107  4112  4119  4121  4123  4244  4257  4266  4275  4284
          4299  4308  4317  4326  4331  4338  4340  4342  4602  4604  4606  4890  4894  4908
          4910   4915   4921   4926   4940   5037   5039   5041   5164   5166   5168   5284   5286   5288
          5447   5449   5451   5453   5455   5457   5462   5468   5473   5479   5484   5490   5495   5501          SEQ 0685
          5506   5512   5517   5523   5541   5558   5560   5718   5720   5722   5724   5726   5728   5733
          5739   5744   5750   5755   5761   5766   5772   5777   5783   5788   5794   5812   5829   5831
          6032   6036   6041   6045   6050   6054   6059   6061   6066   6070   6074   6103   6107   6112
          6114   6291   6295   6300   6304   6309   6313   6318   6320   6325   6329   6333   6367   6373
          6378   6380   6555   6559   6564   6568   6573   6577   6582   6584   6589   6593   6597   6626
          6630   6635   6637   6826   6836   6840   6955   6970   6978   6988   6995   7003   7013   7020
          7028   7038   7045   7053   7055   7062   7381   7396   7404   7414   7421   7429   7439   7446
          7454   7456   7461   7465   7473   7483   7490   7498   7500   7507
.RJZ      1087   1649   2202   2373   3526   3680   4431   4684   4760   4885
.RLDCT    1077   3266   3275   3808   4029   4248   4898   4902   4928   5464   5475   5486   5497   5508
          5519   5533   5735   5746   5757   5768   5779   5790   5804   6080   6093   6339   6353   6359
          6603   6616   6828   6957   6961   7383   7387
.RLDLM    2956
.RMGC     1077   1085   2007   2009   2013   2018   2022   2024   2108   2196   2200   2280   2284   2363
          2365   2369   2448   2532   2632   2636   2778   2780   2782   2784   2786   2788   2790   2792
          2794   2796   2798   2800   2802   2804   2806   2808   2810   2812   2814   2816   2818   2820
          2822   2824   2826   2828   2830   2832   2834   2836   2838   2840   2842   2844   2846   2848
          2850   2852   2854   2856   2858   2860   2862   2864   2866   2868   2870   2872   2956   2958
          2960   3136   3138   3288   3294   3526   3680   3815   3819   3831   3833   3837   3842   3846
          3864   3866   3870   3875   3879   4036   4040   4052   4054   4058   4063   4067   4085   4087
          4091   4096   4100   4255   4259   4271   4273   4277   4282   4286   4304   4306   4310   4315
          4319   4896   4898   4902   4906   4910   4915   4934   4936   4942   5447   5449   5451   5453
          5455   5457   5468   5479   5490   5501   5512   5523   5537   5539   5558   5718   5720   5722
          5724   5726   5728   5739   5750   5761   5772   5783   5794   5808   5810   5829   6034   6036
          6043   6045   6052   6054   6061   6072   6097   6099   6105   6112   6293   6295   6302   6304
          6311   6313   6320   6331   6357   6359   6363   6369   6371   6378   6557   6559   6566   6568
          6575   6577   6584   6595   6620   6622   6628   6635   6826   6832   6834   6838   6955   6968
          6972   6978   6984   6986   6990   6993   6997   7003   7009   7011   7015   7018   7022   7028
          7034   7036   7040   7043   7047   7053   7055   7060   7381   7394   7398   7404   7410   7412
          7416   7419   7423   7429   7435   7437   7441   7444   7448   7454   7456   7463   7467   7473
          7479   7481   7485   7488   7492   7498   7500   7505
.ROENA    1085   2007   2024   2363   2780   2784   2788   2792   2796   2800   2804   2808   2812   2816
          2820   2824   2828   2832   2836   2840   2844   2848   2852   2856   2860   2864   2868   2872
          2956   2960   3138   3288   3294   3526   3680   3831   3864   4052   4085   4271   4304   4934
          5537   5558   5808   5829   6097   6112   6357   6369   6378   6620   6635   6832   6984   7009
          7034   7060   7410   7435   7479   7505
.ROR      1077   1085   2013   2022   2024   2778   2780   2782   2784   2786   2788   2790   2792   2794
          2796   2798   2800   2802   2804   2806   2808   2810   2812   2814   2816   2818   2820   2822
          2824   2826   2828   2830   2832   2834   2836   2838   2840   2842   2844   2846   2848   2850
          2852   2854   2856   2858   2860   2862   2864   2866   2868   2870   2872   2956   2958   2960
          3136   3138   3270   3279   3288   3819   3831   3837   3846   3879   4040   4052   4058   4067
          4100   4259   4271   4277   4286   4319   4896   4898   4900   4902   4904   4910   4926   4932
          4934   4942   5447   5449   5451   5453   5455   5457   5468   5479   5490   5501   5512   5523
          5535   5537   5549   5551   5558   5718   5720   5722   5724   5726   5728   5739   5750   5761
          5772   5783   5794   5806   5808   5820   5822   5829   6034   6036   6043   6045   6052   6054
          6061   6072   6080   6082   6095   6097   6105   6112   6293   6295   6302   6304   6311   6313
          6320   6331   6339   6341   6355   6357   6359   6369   6378   6557   6559   6566   6568   6575
          6577   6584   6595   6603   6605   6618   6620   6628   6635   6826   6830   6832   6838   6955
          6959   6963   6972   6978   6997   7003   7009   7022   7028   7047   7053   7055   7060   7381
          7385   7389   7398   7404   7423   7429   7435   7448   7454   7456   7467   7473   7479   7492
          7498   7500   7505
```

```
.RPLUS    3268   3272   3277   3281   3806   3810   4027   4031   4246   4250   4917   5464   5475   5486
          5497   5508   5519   5735   5746   5757   5768   5779   5790
.RRDLM    2958   3526   3680
.RRPCT    1079   3270   3279   3810   4031   4250   4900   4904   4930   5535   5549   5806   5820   6082
          6095   6341   6355   6361   6605   6618   6830   6959   6963   7385   7389
.RSOA     1085   2013   2024   2363   2780   2784   2788   2792   2796   2800   2804   2808   2812   2816
          2820   2824   2828   2832   2836   2840   2844   2848   2852   2856   2860   2864   2868   2872
          2956   2960   3138   3254   3256   3258   3260   3263   3288   3831   3837   3864   4052   4058
          4085   4271   4277   4304   4915   4926   5462   5473   5484   5495   5506   5517   5531   5535
          5547   5549   5553   5558   5733   5744   5755   5766   5777   5788   5802   5806   5818   5820
          5824   5829   6080   6091   6095   6112   6339   6351   6355   6369   6378   6603   6614   6618
          6635   6830   6832   6838   6984   7009   7034   7060   7410   7435   7479   7505
.RSOB     2007   3268   3270   3272   3277   3279   3281   3283   3806   4027   4246   4900   4904   4917
          4934   6082   6341   6361   6605   6959   6963   7385   7389
.RSAB     3810   3824   3851   3884   4031   4045   4072   4105   4250   4264   4291   4324   4932   5464
          5475   5486   5497   5508   5519   5537   5551   5735   5746   5757   5768   5779   5790   5808
          5822   6097   6357   6620   6974   6976   6999   7001   7024   7026   7049   7051   7400   7402
          7425   7427   7450   7452   7469   7471   7494   7496
.RSDO     1077   2005   2022   2778   2782   2786   2790   2794   2798   2802   2806   2810   2814   2818
          2822   2826   2830   2834   2838   2842   2846   2850   2854   2858   2862   2866   2870   2954
          2958   3136   3804   3819   3846   3879   4025   4040   4067   4100   4244   4259   4286   4319
          4896   4898   4902   4910   4942   5447   5449   5451   5453   5455   5457   5468   5479   5490
          5501   5512   5523   5718   5720   5722   5724   5726   5728   5739   5750   5761   5772   5783
          5794   6034   6036   6043   6045   6052   6054   6061   6072   6105   6293   6295   6302   6304
          6311   6313   6320   6331   6359   6557   6559   6566   6568   6575   6577   6584   6595   6628
          6826   6955   6972   6978   6997   7003   7022   7028   7047   7053   7055   7381   7398   7404
          7423   7429   7448   7454   7456   7467   7473   7492   7498   7500
.RSELE    1077   1085   2007   2009   2013   2018   2022   2024   2108   2196   2200   2280   2284   2363
          2365   2369   2448   2532   2632   2636   2780   2784   2788   2792   2796   2800   2804   2808
          2812   2816   2820   2824   2828   2832   2836   2840   2844   2848   2852   2856   2860   2864
          2868   2872   2960   3136   3138   3288   3294   3526   3680   3815   3819   3831   3833   3837
          3842   3846   3864   3866   3870   3875   3879   4036   4040   4052   4054   4058   4063   4067
          4085   4087   4091   4096   4100   4255   4259   4271   4273   4277   4282   4286   4304   4306
          4310   4315   4319   4896   4906   4910   4915   4934   4936   4942   5537   5539   5558   5808
          5810   5829   6072   6097   6099   6105   6112   6331   6357   6363   6369   6371   6378   6595
          6620   6622   6628   6635   6832   6834   6838   6968   6972   6984   6986   6990   6993   6997
          7009   7011   7015   7018   7022   7034   7036   7040   7043   7047   7060   7394   7398   7410
          7412   7416   7419   7423   7435   7437   7441   7444   7448   7463   7467   7479   7481   7485
          7488   7492   7505
.RSKCN    2778   2782   2786   2790   2794   2798   2802   2806   2810   2814   2818   2822   2826   2830
          2834   2838   2842   2846   2850   2854   2858   2862   2866   2870   4898   4902   5447   5449
          5451   5453   5455   5457   5468   5479   5490   5501   5512   5523   5718   5720   5722   5724
          5726   5728   5739   5750   5761   5772   5783   5794   6034   6036   6043   6045   6052   6054
          6061   6293   6295   6302   6304   6311   6313   6320   6359   6557   6559   6566   6568   6575
          6577   6584   6826   6955   6978   7003   7028   7053   7055   7381   7404   7429   7454   7456
          7473   7498   7500
.RXNOR    2007   3263   3283
.RXOR     7026   7051   7471
```

```
ATABLE   2720   2722   2724   2726   2728   2730   2732   2734   2736   2738   2740   2742   2744   2746
         2748   2750   2752   2754   2756   2758   2760   2762   2764   2766   2768   2943   2945
CALC     1071   1073   1075   1077   1079   1081   1083   1085   1087   1649   2005   2007   2009   2011
         2013   2018   2020   2022   2024   2026   2108   2110   2112   2196   2198   2200   2202   2280
         2282   2284   2286   2363   2365   2367   2369   2371   2373   2448   2450   2452   2532   2534
         2536   2632   2634   2636   2638   2778   2780   2782   2784   2786   2788   2790   2792   2794
         2796   2798   2800   2802   2804   2806   2808   2810   2812   2814   2816   2818   2820   2822
         2824   2826   2828   2830   2832   2834   2836   2838   2840   2842   2844   2846   2848   2850
         2852   2854   2856   2858   2860   2862   2864   2866   2868   2870   2872   2954   2956   2958
         2960   3130   3132   3134   3136   3138   3140   3254   3256   3258   3260   3263   3266   3268
         3270   3272   3275   3277   3279   3281   3283   3288   3290   3292   3294   3296   3370   3372
         3374   3446   3448   3450   3526   3596   3598   3600   3680   3804   3806   3808   3810   3815
         3817   3819   3824   3826   3831   3833   3835   3837   3842   3844   3846   3851   3857   3859
         3864   3866   3868   3870   3875   3877   3879   3884   3886   3891   3898   3900   3902   4025
         4027   4029   4031   4036   4038   4040   4045   4047   4052   4054   4056   4058   4063   4065
         4067   4072   4078   4080   4085   4087   4089   4091   4096   4098   4100   4105   4107   4112
         4119   4121   4123   4244   4246   4248   4250   4255   4257   4259   4264   4266   4271   4273
         4275   4277   4282   4284   4286   4291   4297   4299   4304   4306   4308   4310   4315   4317
         4319   4324   4326   4331   4338   4340   4342   4431   4602   4604   4606   4684   4760   4885
         4890   4892   4894   4896   4898   4900   4902   4904   4906   4908   4910   4915   4917   4919
         4921   4926   4928   4930   4932   4934   4936   4938   4940   4942   4944   5037   5039   5041
         5164   5166   5168   5284   5286   5288   5447   5449   5451   5453   5455   5457   5462   5464
         5466   5468   5473   5475   5477   5479   5484   5486   5488   5490   5495   5497   5499   5501
         5506   5508   5510   5512   5517   5519   5521   5523   5531   5533   5535   5537   5539   5541
         5547   5549   5551   5553   5558   5560   5718   5720   5722   5724   5726   5728   5733   5735
         5737   5739   5744   5746   5748   5750   5755   5757   5759   5761   5766   5768   5770   5772
         5777   5779   5781   5783   5788   5790   5792   5794   5802   5804   5806   5808   5810   5812
         5818   5820   5822   5824   5829   5831   6032   6034   6036   6041   6043   6045   6050   6052
         6054   6059   6061   6066   6068   6070   6072   6074   6080   6082   6084   6091   6093   6095
         6097   6099   6101   6103   6105   6107   6112   6114   6291   6293   6295   6300   6302   6304
         6309   6311   6313   6318   6320   6325   6327   6329   6331   6333   6339   6341   6343   6351
         6353   6355   6357   6359   6361   6363   6365   6367   6369   6371   6373   6378   6380   6555
         6557   6559   6564   6566   6568   6573   6575   6577   6582   6584   6589   6591   6593   6595
         6597   6603   6605   6607   6614   6616   6618   6620   6622   6624   6626   6628   6630   6635
         6637   6826   6828   6830   6832   6834   6836   6838   6840   6955   6957   6959   6961   6963
         6968   6970   6972   6974   6976   6978   6984   6986   6988   6990   6993   6995   6997   6999
         7001   7003   7009   7011   7013   7015   7018   7020   7022   7024   7026   7028   7034   7036
         7038   7040   7043   7045   7047   7049   7051   7053   7055   7060   7062   7381   7383   7385
         7387   7389   7394   7396   7398   7400   7402   7404   7410   7412   7414   7416   7419   7421
         7423   7425   7427   7429   7435   7437   7439   7441   7444   7446   7448   7450   7452   7454
         7456   7461   7463   7465   7467   7469   7471   7473   7479   7481   7483   7485   7488   7490
         7492   7494   7496   7498   7500   7505   7507
CONCAT   1071   1073   1075   1077   1079   1081   1083   1085   1087   1649   2005   2007   2009   2011
         2013   2018   2020   2022   2024   2026   2108   2110   2112   2196   2198   2200   2202   2280
         2282   2284   2286   2363   2365   2367   2369   2371   2373   2448   2450   2452   2532   2534
         2536   2632   2634   2636   2638   2778   2780   2782   2784   2786   2788   2790   2792   2794
         2796   2798   2800   2802   2804   2806   2808   2810   2812   2814   2816   2818   2820   2622
         2824   2826   2828   2830   2832   2834   2836   2838   2840   2842   2844   2846   2848   2850
         2852   2854   2856   2858   2860   2862   2864   2866   2868   2870   2872   2954   2956   2958
         2960   3130   3132   3134   3136   3138   3140   3254   3256   3258   3260   3263   3266   3268
         3270   3272   3275   3277   3279   3281   3283   3288   3290   3292   3294   3295   3370   3372
         3374   3446   3448   3450   3526   3596   3598   3600   3680   3804   3806   3808   3810   3815
         3817   3819   3824   3826   3831   3833   3835   3837   3842   3844   3846   3851   3857   3859
```

```
             3864    3866    3868    3870    3875    3877    3879    3884    3886    3891    3898    3900    3902    4025        SEQ 0688
             4027    4029    4031    4036    4038    4040    4045    4047    4052    4054    4056    4058    4063    4065
             4067    4072    4078    4080    4085    4087    4089    4091    4096    4098    4100    4105    4107    4112
             4119    4121    4123    4244    4246    4248    4250    4255    4257    4259    4264    4266    4271    4273
             4275    4277    4282    4284    4286    4291    4297    4299    4304    4306    4308    4310    4315    4317
             4319    4324    4326    4331    4338    4340    4342    4431    4602    4604    4606    4684    4760    4885
             4890    4892    4894    4896    4898    4900    4902    4904    4906    4908    4910    4915    4917    4919
             4921    4926    4928    4930    4932    4934    4936    4938    4940    4942    4944    5037    5039    5041
             5164    5166    5168    5284    5286    5288    5447    5449    5451    5453    5455    5457    5462    5464
             5466    5468    5473    5475    5477    5479    5484    5486    5488    5490    5495    5497    5499    5501
             5506    5508    5510    5512    5517    5519    5521    5523    5531    5533    5535    5537    5539    5541
             5547    5549    5551    5553    5558    5560    5718    5720    5722    5724    5726    5728    5733    5735
             5737    5739    5744    5746    5748    5750    5755    5757    5759    5761    5766    5768    5770    5772
             5777    5779    5781    5783    5788    5790    5792    5794    5802    5804    5806    5808    5810    5812
             5818    5820    5822    5824    5829    5831    6032    6034    6036    6041    6043    6045    6050    6052
             6054    6059    6061    6066    6068    6070    6072    6074    6080    6082    6084    6091    6093    6095
             6097    6099    6101    6103    6105    6107    6112    6114    6291    6293    6295    6300    6302    6304
             6309    6311    6313    6318    6320    6325    6327    6329    6331    6333    6339    6341    6343    6351
             6353    6355    6357    6359    6361    6363    6365    6367    6369    6371    6373    6378    6380    6555
             6557    6559    6564    6566    6568    6573    6575    6577    6582    6584    6589    6591    6593    6595
             6597    6603    6605    6607    6614    6616    6618    6620    6622    6624    6626    6628    6630    6635
             6637    6826    6828    6830    6832    6834    6836    6838    6840    6955    6957    6959    6961    6963
             6968    6970    6972    6974    6976    6978    6984    6986    6988    6990    6993    6995    6997    6999
             7001    7003    7009    7011    7013    7015    7018    7020    7022    7024    7026    7028    7034    7036
             7038    7040    7043    7045    7047    7049    7051    7053    7055    7060    7062    7381    7383    7385
             7387    7389    7394    7396    7398    7400    7402    7404    7410    7412    7414    7416    7419    7421
             7423    7425    7427    7429    7435    7437    7439    7441    7444    7446    7448    7450    7452    7454
             7456    7461    7463    7465    7467    7469    7471    7473    7479    7481    7483    7485    7488    7490
             7492    7494    7496    7498    7500    7505    7507
FIELD        1071    1073    1075    1077    1079    1081    1083    1085    1087    1649    2005    2007    2009    2011
             2013    2018    2020    2022    2024    2026    2108    2110    2112    2196    2198    2200    2202    2280
             2282    2284    2286    2363    2365    2367    2369    2371    2373    2448    2450    2452    2532    2534
             2536    2632    2634    2636    2638    2778    2780    2782    2784    2786    2788    2790    2792    2794
             2796    2798    2800    2802    2804    2806    2808    2810    2812    2814    2816    2818    2820    2822
             2824    2826    2828    2830    2832    2834    2836    2838    2840    2842    2844    2846    2848    2850
             2852    2854    2856    2858    2860    2862    2864    2866    2868    2870    2872    2954    2956    2958
             2960    3130    3132    3134    3136    3138    3140    3254    3256    3258    3260    3263    3266    3268
             3270    3272    3275    3277    3279    3281    3283    3288    3290    3292    3294    3296    3370    3372
             3374    3446    3448    3450    3526    3596    3598    3600    3680    3804    3806    3808    3810    3815
             3817    3819    3824    3826    3831    3833    3835    3837    3842    3844    3846    3851    3857    3859
             3864    3866    3868    3870    3875    3877    3879    3884    3886    3891    3898    3900    3902    4025
             4027    4029    4031    4036    4038    4040    4045    4047    4052    4054    4056    4058    4063    4065
             4067    4072    4078    4080    4085    4087    4089    4091    4096    4098    4100    4105    4107    4112
             4119    4121    4123    4244    4246    4248    4250    4255    4257    4259    4264    4266    4271    4273
             4275    4277    4282    4284    4286    4291    4297    4299    4304    4306    4308    4310    4315    4317
             4319    4324    4326    4331    4338    4340    4342    4431    4602    4604    4606    4684    4760    4885
             4890    4892    4894    4896    4898    4900    4902    4904    4906    4908    4910    4915    4917    4919
             4921    4926    4928    4930    4932    4934    4936    4938    4940    4942    4944    5037    5039    5041
             5164    5166    5168    5284    5286    5288    5447    5449    5451    5453    5455    5457    5462    5464
             5466    5468    5473    5475    5477    5479    5484    5486    5488    5490    5495    5497    5499    5501
             5506    5508    5510    5512    5517    5519    5521    5523    5531    5533    5535    5537    5539    5541
             5547    5549    5551    5553    5558    5560    5718    5720    5722    5724    5726    5728    5733    5735
             5737    5739    5744    5746    5748    5750    5755    5757    5759    5761    5766    5768    5770    5772
```

```
          5777   5779   5781   5783   5788   5790   5792   5794   5802   5804   5806   5808   5810   5812
          5818   5820   5822   5824   5829   5831   6032   6034   6036   6041   6043   6045   6050   6052
          6054   6059   6061   6066   6068   6070   6072   6074   6080   6082   6084   6091   6093   6095
          6097   6099   6101   6103   6105   6107   6112   6114   6291   6293   6295   6300   6302   6304
          6309   6311   6313   6318   6320   6325   6327   6329   6331   6333   6339   6341   6343   6351
          6353   6355   6357   6359   6361   6363   6365   6367   6369   6371   6373   6378   6380   6555
          6557   6559   6564   6566   6568   6573   6575   6577   6582   6584   6589   6591   6593   6595
          6597   6603   6605   6607   6614   6616   6618   6620   6622   6624   6626   6628   6630   6635
          6637   6826   6828   6830   6832   6834   6836   6838   6840   6955   6957   6959   6961   6963
          6968   6970   6972   6974   6976   6978   6984   6986   6988   6990   6993   6995   6997   6999
          7001   7003   7009   7011   7013   7015   7018   7020   7022   7024   7026   7028   7034   7036
          7038   7040   7043   7045   7047   7049   7051   7053   7055   7060   7062   7381   7383   7385
          7387   7389   7394   7396   7398   7400   7402   7404   7410   7412   7414   7416   7419   7421
          7423   7425   7427   7429   7435   7437   7439   7441   7444   7446   7448   7450   7452   7454
          7456   7461   7463   7465   7467   7469   7471   7473   7479   7481   7483   7485   7488   7490
          7492   7494   7496   7498   7500   7505   7507
GET       7659   7660
GO          73     76     77    133    134    146    150    153    235    241    242    252    255    258
           335    341    352    354    359    361    438    444    445    456    460    462    538    544
           549    550    602    616    618    619    635    636    774    781    786    787    802    807
           808    863    866    868    869    918    923    925    927    928    989    995    997   1008
          1030   1031   1066   1125   1135   1137   1139   1140   1153   1205   1211   1215   1217   1219
          1220   1222   1223   1276   1291   1293   1295   1296   1427   1430   1431   1476   1481   1482
          1529   1533   1535   1537   1538   1540   1541   1596   1598   1606   1609   1611   1613   1627
          1628   1693   1705   1707   1709   1710   1879   1892   1894   1896   1897   1961   1963   1974
          1985   1986   2065   2067   2079   2088   2089   2151   2153   2165   2174   2238   2240   2252
          2260   2321   2323   2335   2343   2405   2407   2419   2427   2428   2483   2489   2491   2503
          2511   2512   2572   2574   2586   2594   2608   2609   2682   2684   2690   2696   2911   2913
          2919   2924   2996   2999   3001   3002   3003   3053   3055   3066   3068   3070   3076   3077
          3175   3177   3188   3192   3194   3328   3330   3341   3349   3350   3405   3407   3418   3426
          3484   3486   3497   3505   3555   3557   3568   3576   3633   3635   3646   3654   3723   3725
          3736   3744   3758   3759   3772   3773   3944   3946   3957   3965   3979   3980   3993   3994
          4163   4165   4176   4184   4198   4199   4212   4213   4390   4392   4403   4411   4471   4477
          4480   4485   4493   4494   4502   4503   4524   4561   4563   4574   4582   4641   4643   4654
          4663   4664   4717   4719   4730   4739   4740   4801   4807   4809   4818   4819   4820   4821
          4822   4828   4830   4843   4847   4851   4981   4987   4989   4995   4996   4997   4998   5007
          5016   5019   5085   5091   5093   5104   5105   5106   5118   5134   5207   5213   5215   5221
          5222   5223   5232   5248   5258   5339   5345   5347   5358   5359   5610   5616   5618   5629
          5630   5904   5906   5914   5922   5923   6005   6007   6013   6162   6164   6172   6180   6181
          6264   6266   6272   6426   6428   6436   6444   6445   6522   6524   6530   6687   6689   6696
          6699   6713   6714   6728   6729   6744   6746   6747   6762   6763   6779   6781   6782   6783
          6802   6887   6889   6896   6902   6912   6913   6933   7105   7111   7114   7128   7129   7143
          7144   7159   7160   7175   7176   7190   7191   7210   7278   7280   7287   7293   7303   7304
          7320   7325   7335   7336   7356   7530   7577   7580   7587   7588
ITABLE    5377   5378   5379   5380   5381   5382   5383   5384   5385   5386   5387   5648   5649   5650
          5651   5652   5653   5654   5655   5656   5657   5658   5941   5942   5944   5945   5946   5947
          5948   5950   5951   5952   5953   5954   5955   5956   5957   6199   6200   6202   6203   6204
          6205   6206   6208   6209   6210   6211   6212   6213   6214   6215   6463   6464   6466   6467
          6468   6469   6470   6472   6473   6474   6475   6476   6477   6478   6479
MFLD      1071#  1071   1073#  1073   1075#  1075   1077#  1077   1079#  1079   1081#  1081   1083#  1083
          1085#  1085   1087#  1087   1649#  1649   2005#  2005   2007#  2007   2009#  2009   2011#  2011
          2013#  2013   2018#  2018   2020#  2020   2022#  2022   2024#  2024   2026#  2026   2108#  2108
          2110#  2110   2112#  2112   2196#  2196   2198#  2198   2200#  2200   2202#  2202   2280#  2280
```

```
677
      2282#  2282   2284#  2284   2286#  2286   2363#  2363   2365#  2365   2367#  2367   2369#  2369      SEQ 0690
      2371#  2371   2373#  2373   2448#  2448   2450#  2450   2452#  2452   2532#  2532   2534#  2534
      2536#  2536   2632#  2632   2634#  2634   2636#  2636   2638#  2638   2778#  2778   2780#  2780
      2782#  2782   2784#  2784   2786#  2786   2788#  2788   2790#  2790   2792#  2792   2794#  2794
      2796#  2796   2798#  2798   2800#  2800   2802#  2802   2804#  2804   2806#  2806   2808#  2808
      2810#  2810   2812#  2812   2814#  2814   2816#  2816   2818#  2818   2820#  2820   2822#  2822
      2824#  2824   2826#  2826   2828#  2828   2830#  2830   2832#  2832   2834#  2834   2836#  2836
      2838#  2838   2840#  2840   2842#  2842   2844#  2844   2846#  2846   2848#  2848   2850#  2850
      2852#  2852   2854#  2854   2856#  2856   2858#  2858   2860#  2860   2862#  2862   2864#  2864
      2866#  2866   2868#  2868   2870#  2870   2872#  2872   2954#  2954   2956#  2956   2958#  2958
      2960#  2960   3130#  3130   3132#  3132   3134#  3134   3136#  3136   3138#  3138   3140#  3140
      3254#  3254   3256#  3256   3258#  3258   3260#  3260   3263#  3263   3266#  3266   3268#  3268
      3270#  3270   3272#  3272   3275#  3275   3277#  3277   3279#  3279   3281#  3281   3283#  3283
      3288#  3288   3290#  3290   3292#  3292   3294#  3294   3296#  3296   3370#  3370   3372#  3372
      3374#  3374   3446#  3446   3448#  3448   3450#  3450   3526#  3526   3596#  3596   3598#  3598
      3600#  3600   3680#  3680   3804#  3804   3806#  3806   3808#  3808   3810#  3810   3815#  3815
      3817#  3817   3819#  3819   3824#  3824   3826#  3826   3831#  3831   3833#  3833   3835#  3835
      3837#  3837   3842#  3842   3844#  3844   3846#  3846   3851#  3851   3857#  3857   3859#  3859
      3864#  3864   3866#  3866   3868#  3868   3870#  3870   3875#  3875   3877#  3877   3879#  3879
      3884#  3884   3886#  3886   3891#  3891   3898#  3898   3900#  3900   3902#  3902   4025#  4025
      4027#  4027   4029#  4029   4031#  4031   4036#  4036   4038#  4038   4040#  4040   4045#  4045
      4047#  4047   4052#  4052   4054#  4054   4056#  4056   4058#  4058   4063#  4063   4065#  4065
      4067#  4067   4072#  4072   4078#  4078   4080#  4080   4085#  4085   4087#  4087   4089#  4089
      4091#  4091   4096#  4096   4098#  4098   4100#  4100   4105#  4105   4107#  4107   4112#  4112
      4119#  4119   4121#  4121   4123#  4123   4244#  4244   4246#  4246   4248#  4248   4250#  4250
      4255#  4255   4257#  4257   4259#  4259   4264#  4264   4266#  4266   4271#  4271   4273#  4273
      4275#  4275   4277#  4277   4282#  4282   4284#  4284   4286#  4286   4291#  4291   4297#  4297
      4299#  4299   4304#  4304   4306#  4306   4308#  4308   4310#  4310   4315#  4315   4317#  4317
      4319#  4319   4324#  4324   4326#  4326   4331#  4331   4338#  4338   4340#  4340   4342#  4342
      4431#  4431   4602#  4602   4604#  4604   4606#  4606   4684#  4684   4760#  4760   4885#  4885
      4890#  4890   4892#  4892   4894#  4894   4896#  4896   4898#  4898   4900#  4900   4902#  4902
      4904#  4904   4906#  4906   4908#  4908   4910#  4910   4915#  4915   4917#  4917   4919#  4919
      4921#  4921   4926#  4926   4928#  4928   4930#  4930   4932#  4932   4934#  4934   4936#  4936
      4938#  4938   4940#  4940   4942#  4942   4944#  4944   5037#  5037   5039#  5039   5041#  5041
      5164#  5164   5166#  5166   5168#  5168   5284#  5284   5286#  5286   5288#  5288   5447#  5447
      5449#  5449   5451#  5451   5453#  5453   5455#  5455   5457#  5457   5462#  5462   5464#  5464
      5466#  5466   5468#  5468   5473#  5473   5475#  5475   5477#  5477   5479#  5479   5484#  5484
      5486#  5486   5488#  5488   5490#  5490   5495#  5495   5497#  5497   5499#  5499   5501#  5501
      5506#  5506   5508#  5508   5510#  5510   5512#  5512   5517#  5517   5519#  5519   5521#  5521
      5523#  5523   5531#  5531   5533#  5533   5535#  5535   5537#  5537   5539#  5539   5541#  5541
      5547#  5547   5549#  5549   5551#  5551   5553#  5553   5558#  5558   5560#  5560   5718#  5718
      5720#  5720   5722#  5722   5724#  5724   5726#  5726   5728#  5728   5733#  5733   5735#  5735
      5737#  5737   5739#  5739   5744#  5744   5746#  5746   5748#  5748   5750#  5750   5755#  5755
      5757#  5757   5759#  5759   5761#  5761   5766#  5766   5768#  5768   5770#  5770   5772#  5772
      5777#  5777   5779#  5779   5781#  5781   5783#  5783   5788#  5788   5790#  5790   5792#  5792
      5794#  5794   5802#  5802   5804#  5804   5806#  5806   5808#  5808   5810#  5810   5812#  5812
      5818#  5818   5820#  5820   5822#  5822   5824#  5824   5829#  5829   5831#  5831   6032#  6032
      6034#  6034   6036#  6036   6041#  6041   6043#  6043   6045#  6045   6050#  6050   6052#  6052
      6054#  6054   6059#  6059   6061#  6061   6066#  6066   6068#  6068   6070#  6070   6072#  6072
      6074#  6074   6080#  6080   6082#  6082   6084#  6084   6091#  6091   6093#  6093   6095#  6095
      6097#  6097   6099#  6099   6101#  6101   6103#  6103   6105#  6105   6107#  6107   6112#  6112
      6114#  6114   6291#  6291   6293#  6293   6295#  6295   6300#  6300   6302#  6302   6304#  6304
      6309#  6309   6311#  6311   6313#  6313   6318#  6318   6320#  6320   6325#  6325   6327#  6327

678   6329#  6329   6331#  6331   6333#  6333   6339#  6339   6341#  6341   6343#  6343   6351#  6351
```

```
                6329#  6329   6331#  6331   6333#  6333   6339#  6339   6341#  6341   6343#  6343   6351#  6351      SEQ 0691
                6353#  6353   6355#  6355   6357#  6357   6359#  6359   6361#  6361   6363#  6363   6365#  6365
                6367#  6367   6369#  6369   6371#  6371   6373#  6373   6378#  6378   6380#  6380   6555#  6555
                6557#  6557   6559#  6559   6564#  6564   6566#  6566   6568#  6568   6573#  6573   6575#  6575
                6577#  6577   6582#  6582   6584#  6584   6589#  6589   6591#  6591   6593#  6593   6595#  6595
                6597#  6597   6603#  6603   6605#  6605   6607#  6607   6614#  6614   6616#  6616   6618#  6618
                6620#  6620   6622#  6622   6624#  6624   6626#  6626   6628#  6628   6630#  6630   6635#  6635
                6637#  6637   6826#  6826   6828#  6828   6830#  6830   6832#  6832   6834#  6834   6836#  6836
                6838#  6838   6840#  6840   6955#  6955   6957#  6957   6959#  6959   6961#  6961   6963#  6963
                6968#  6968   6970#  6970   6972#  6972   6974#  6974   6976#  6976   6978#  6978   6984#  6984
                6986#  6986   6988#  6988   6990#  6990   6993#  6993   6995#  6995   6997#  6997   6999#  6999
                7001#  7001   7003#  7003   7009#  7009   7011#  7011   7013#  7013   7015#  7015   7018#  7018
                7020#  7020   7022#  7022   7024#  7024   7026#  7026   7028#  7028   7034#  7034   7036#  7036
                7038#  7038   7040#  7040   7043#  7043   7045#  7045   7047#  7047   7049#  7049   7051#  7051
                7053#  7053   7055#  7055   7060#  7060   7062#  7062   7381#  7381   7383#  7383   7385#  7385
                7387#  7387   7389#  7389   7394#  7394   7396#  7396   7398#  7398   7400#  7400   7402#  7402
                7404#  7404   7410#  7410   7412#  7412   7414#  7414   7416#  7416   7419#  7419   7421#  7421
                7423#  7423   7425#  7425   7427#  7427   7429#  7429   7435#  7435   7437#  7437   7439#  7439
                7441#  7441   7444#  7444   7446#  7446   7448#  7448   7450#  7450   7452#  7452   7454#  7454
                7456#  7456   7461#  7461   7463#  7463   7465#  7465   7467#  7467   7469#  7469   7471#  7471
                7473#  7473   7479#  7479   7481#  7481   7483#  7483   7485#  7485   7488#  7488   7490#  7490
                7492#  7492   7494#  7494   7496#  7496   7498#  7498   7500#  7500   7505#  7505   7507#  7507
MWORD           1071   1073   1075   1077   1079   1081   1083   1085   1087   1649   2005   2007   2009   2011
                2013   2018   2020   2022   2024   2026   2108   2110   2112   2196   2198   2200   2202   2280
                2282   2284   2286   2363   2365   2367   2369   2371   2373   2448   2450   2452   2532   2534
                2536   2632   2634   2636   2638   2778   2780   2782   2784   2786   2788   2790   2792   2794
                2796   2798   2800   2802   2804   2806   2808   2810   2812   2814   2816   2818   2820   2822
                2824   2826   2828   2830   2832   2834   2836   2838   2840   2842   2844   2846   2848   2850
                2852   2854   2856   2858   2860   2862   2864   2866   2868   2870   2872   2954   2956   2958
                2960   3130   3132   3134   3136   3138   3140   3254   3256   3258   3260   3263   3266   3268
                3270   3272   3275   3277   3279   3281   3283   3288   3290   3292   3294   3296   3370   3372
                3374   3446   3448   3450   3526   3596   3598   3600   3680   3804   3806   3808   3810   3815
                3817   3819   3824   3826   3831   3833   3835   3837   3842   3844   3846   3851   3857   3859
                3864   3866   3868   3870   3875   3877   3879   3884   3886   3891   3898   3900   3902   4025
                4027   4029   4031   4036   4038   4040   4045   4047   4052   4054   4056   4058   4063   4065
                4067   4072   4078   4080   4085   4087   4089   4091   4096   4098   4100   4105   4107   4112
                4119   4121   4123   4244   4246   4248   4250   4255   4257   4259   4264   4266   4271   4273
                4275   4277   4282   4284   4286   4291   4297   4299   4304   4306   4308   4310   4315   4317
                4319   4324   4326   4331   4338   4340   4342   4431   4602   4604   4606   4684   4760   4885
                4890   4892   4894   4896   4898   4900   4902   4904   4906   4908   4910   4915   4917   4919
                4921   4926   4928   4930   4932   4934   4936   4938   4940   4942   4944   5037   5039   5041
                5164   5166   5168   5284   5286   5288   5447   5449   5451   5453   5455   5457   5462   5464
                5466   5468   5473   5475   5477   5479   5484   5486   5488   5490   5495   5497   5499   5501
                5506   5508   5510   5512   5517   5519   5521   5523   5531   5533   5535   5537   5539   5541
                5547   5549   5551   5553   5558   5560   5718   5720   5722   5724   5726   5728   5733   5735
                5737   5739   5744   5746   5748   5750   5755   5757   5759   5761   5766   5768   5770   5772
                5777   5779   5781   5783   5788   5790   5792   5794   5802   5804   5806   5808   5810   5812
                5818   5820   5822   5824   5829   5831   6032   6034   6036   6041   6043   6045   6050   6052
                6054   6059   6061   6066   6068   6070   6072   6074   6080   6082   6084   6091   6093   6095
                6097   6099   6101   6103   6105   6107   6112   6114   6291   6293   6295   6300   6302   6304
                6309   6311   6313   6318   6320   6325   6327   6329   6331   6333   6339   6341   6343   6351
                6353   6355   6357   6359   6361   6363   6365   6367   6369   6371   6373   6378   6380   6555
                6557   6559   6564   6566   6568   6573   6575   6577   6582   6584   6589   6591   6593   6595
```

```
          6597  6603  6605  6607  6614  6616  6618  6620  6622  6624  6626  6628  6630  6635     SEQ 0692
          6637  6826  6828  6830  6832  6834  6836  6838  6840  6955  6957  6959  6961  6963
          6968  6970  6972  6974  6976  6978  6984  6986  6988  6990  6993  6995  6997  6999
          7001  7003  7009  7011  7013  7015  7018  7020  7022  7024  7026  7028  7034  7036
          7038  7040  7043  7045  7047  7049  7051  7053  7055  7060  7062  7381  7383  7385
          7387  7389  7394  7396  7398  7400  7402  7404  7410  7412  7414  7416  7419  7421
          7423  7425  7427  7429  7435  7437  7439  7441  7444  7446  7448  7450  7452  7454
          7456  7461  7463  7465  7467  7469  7471  7473  7479  7481  7483  7485  7488  7490
          7492  7494  7496  7498  7500  7505  7507
PJRST     7594
PNT1       396
PNT6       664   673
PNTDEC    4877  7631  7644
PNTHW      191   194   197   295   298   399   402   499   502   828  1059  1063  1736  1923
          3119  3125  3243  3249  7544  7550  7563  7569  7612  7615  7618  7621
PNTMSG     183   186   189   192   195   281   284   287   288   290   293   296   391   394
           397   400   485   488   489   492   494   497   500   662   665   671   674   826
           829  1057  1060  1061  1064  1065  1734  1737  1921  1924  1925  3115  3117  3121
          3123  3239  3241  3245  3247  4523  4875  4878  4880  5157  6807  6810  6813  6816
          6819  6822  6938  6941  6944  6948  6949  6950  6951  7215  7218  7221  7224  7227
          7230  7361  7364  7368  7371  7375  7376  7377  7517  7520  7540  7542  7546  7548
          7559  7561  7565  7567  7610  7613  7616  7619  7629  7632  7642  7645  7647  7648
          7649  7650  7651  7652  7653  7654
PNTOCS     182   188   283   286   292   390   487   491   496  5159  7519
PUT       7628  7629
RGET      7659
RPUT      7628
RTN         87   171   184   198   239   274   299   339   379   392   403   442   478   503
           542   560   655   666   675   819   830   879   939   993   998  1050  1067  1163
          1209  1233  1312  1442  1493  1551  1599  1640  1727  1738  1914  1926  1964  1996
          2068  2099  2154  2187  2241  2271  2324  2354  2408  2439  2487  2492  2523  2575
          2620  2685  2716  2914  2939  3014  3056  3099  3126  3178  3223  3250  3331  3361
          3408  3437  3487  3516  3558  3587  3636  3665  3726  3784  3947  4005  4166  4224
          4352  4393  4422  4475  4516  4525  4564  4593  4644  4675  4720  4751  4805  4810
          4867  4881  4985  4990  5027  5089  5094  5149  5160  5211  5216  5269  5343  5348
          5373  5395  5415  5614  5619  5644  5666  5686  5841  5854  5907  5937  5971  5989
          6014  6165  6195  6227  6249  6273  6429  6459  6485  6506  6531  6690  6794  6803
          6890  6925  6934  7202  7211  7281  7348  7357  7521  7531  7551  7570  7581  7593
          7595  7598  7622  7660
SCOPER      81   159   262   367   466   554   626   643   792   813   873   933  1038  1145
          1157  1227  1300  1436  1487  1545  1632  1715  1902  1990  2093  2181  2265  2348
          2433  2517  2599  2614  2709  2932  3008  3086  3210  3355  3431  3510  3581  3659
          3749  3764  3778  3970  3985  3999  4189  4204  4218  4416  4510  4587  4669  4745
          4861  5021  5136  5250  5263  5366  5637  5930  6188  6452  6703  6718  6733  6752
          6767  6787  6918  7118  7133  7149  7165  7181  7195  7309  7341  7592
TMSG       183   189   284   287   293   391   488   492   497   665   674   829  1060  1064
          1737  1924  4873  4878  6805  6808  6811  6814  6817  6820  6936  6939  6942  6945
          7213  7216  7219  7222  7225  7228  7359  7362  7365  7369  7372  7520  7632  7645
          7647  7648  7649  7650  7651  7652  7653  7654
TMSGC      186   192   195   281   288   290   296   394   397   400   485   489   494   500
           662   671   826  1057  1061  1065  1734  1921  1925  3115  3117  3121  3123  3239
          3241  3245  3247  4523  4880  5157  6949  6950  6951  7376  7377  7517  7540  7542
          7546  7548  7559  7561  7565  7567  7610  7613  7616  7619  7629  7642
```

```
                                 TITLE   DFPTA3
                                 SUBTTL  MP Control Module:  2910 Basic Instruction Tests

                                 SALL

                                 SEARCH  DFPTA,MONSYM

                         ; INTERN's

                                 ENTRY   TSTS1,TSTS2,TSTS3,TSTS4,TSTS5,TSTS6,TSTS7,TSTS10
                                 ENTRY   TSTS11,TSTS12,TSTS13,TSTS14,TST[illegible],TSTS16,TSTS17,TSTS20
                                 ENTRY   TSTS21,TSTS22,TSTS23,TSTS24,TSTS[illegible],TSTS26,TSTS27,TSTS30
                                 ENTRY   TSTS31,TSTS32,TSTS33,TSTS34,TSTS35,TSTS36,TSTS37,TSTS40
                                 ENTRY   TSTS41,TSTS42,TSTS43,TSTS44,TSTS45,TSTS46,TSTS47,TSTS50
                                 ENTRY   TSTS51,TSTS52,TSTS53,TSTS54,TSTS55,TSTS56,TSTS57,TSTS60
                                 ENTRY   TSTS61,TSTS62,TSTS63,TSTS64,TSTS65,TSTS66,TSTS67,TSTS70

                         ; EXTERN's located in DFPTA1.MAC (Mainline code..)

                                 EXTERN  TLOAD,CSRPNT,TRACE,TSTSUB,ODELAY,TSLOD1,TSLOD2,SEXEC,SSPNT
                                 EXTERN  CALPAR,DWCRAM,CADDR,CWORDL,CWORDR,PARFLG,UDEBUG,PORTCI,PORTNI

                         ; EXTERN's located in DFPTAI.MAC (Port Device Handling Routines module)

                                 EXTERN  RDLAR,RDCSR,SETLAR,IPACLR,IPASRT,SNEXT,SDATA


                         ;#*********************************************************************
                         ; Z3 - Address for use in DDT
                         ;#*********************************************************************

000000'                  Z3:                                 ; address of 00000'
```

```
                                ;#**********************************************************************
                                ;* Test 1 - JMAP Instruction
                                ;
                                ; Description:  Verify that the JMAP instruction  unconditionally
                                ;               jumps  to  the  address  specified  by the direct
                                ;               inputs, regardless of the TEST inputs.
                                ;
                                ; Procedure:    Clear Port
                                ;               Load test microcode
                                ;
                                ;               Set start address = 0 (Test En/Cond=OFF/OFF)
                                ;               Set MPRUN/SINGLE STEP (execute JMAP)
                                ;               Read LAR, and verify address = 'pattern'
                                ;
                                ;               Set start address = 1 (Test En/Cond=OFF/ON)
                                ;               Set MPRUN/SINGLE STEP (execute JMAP)
                                ;               Read LAR, and verify address = 'pattern'
                                ;
                                ;               Set start address = 2 (Test En/Cond=ON/OFF)
                                ;               Set MPRUN/SINGLE STEP (execute JMAP)
                                ;               Read LAR, and verify address = 'pattern'
                                ;
                                ;               Set start address = 3 (Test En/Cond=ON/ON)
                                ;               Set MPRUN/SINGLE STEP (execute JMAP)
                                ;               Read LAR, and verify address = 'pattern'
                                ;
                                ;               Do for each data pattern:  0's
                                ;                                          1's
                                ;                                          floating 0's
                                ;                                          floating 1's
                                ;
                                ; Failure:       The wrong address was selected.  Either the 2910
                                ;                itself failed or the Jmp Mux was enabled  when it
                                ;                shouldn't have been (so the low order 8 bits came
                                ;                off the MBUS instead of the microword jump field).
                                ;#**********************************************************************

                                ; Test data

000000' 254 00 0 00 000011'     TSTS1:  JRST    TG1                             ; go start test
000001' 240401  000001                  SEQ!MPROC!NDMP!ZSEQ!1                   ; test mask
000002' 000130' 010436'                 T1M,,[ASCIZ ^JMAP Instruction^]
000003' 010442' 010443'                 [EXP MLAST!M7],,[EXP M16,M9,E17,M1,MLAST!M8]
000004' 000000  000141'                 TSTS2                                   ; failure test table
000005' 000000  000226'                 TSTS3                                   ;   ...
000006' 000000  000327'                 TSTS4
000007' 000000  000430'                 TSTS5
000010' 777777  777777                  -1
```

```
                                   ; Start test

000011' 201 00 0 00 000000'        TG1:    MOVEI   Z3                      ; get address of module start
000012' 260 17 0 00 000000*                GO      TRACE                   ; handle trace output

                                   ; Initialization

000013' 260 17 0 00 000000*        TL1:    GO      IPACLR                  ; clear port
000014' 402 00 0 00 000000*                SETZM   TSTSUB                  ; initialize subtest number
000015' 201 06 0 00 000031'                MOVEI   6,TS1                   ; get sstep table address
000016' 201 07 0 00 000046'                MOVEI   7,TS1PAT-1              ; get data pattern table address

                                   ; Loop on single step table entries

000017' 400 15 0 00 000000         TA1:    SETZ    ERFLG,                  ; clear error flag
000020' 260 17 0 00 000000*                GO      SEXEC                   ; execute table entry
000021' 254 00 0 00 000030'                JRST    TX1                     ; end of sstep table
000022' 254 00 0 00 000017'                JRST    TA1                     ; keep looping after call
000023' 474 15 0 00 000000                 SETO    ERFLG,                  ; error - end address incorrect

                                   ; Handle error printouts and scope looping

000024' 027 00 0 00 000126'                SCOPER  MA1                     ; print error message
000025' 254 00 0 00 000013'                JRST    TL1                     ; loop on error
000026' 254 00 0 00 000030'                JRST    TX1                     ; altmode exit
000027' 322 15 0 00 000017'                JUMPE   ERFLG,TA1               ; do next entry (if no error yet)

                                   ; End of test

000030' 263 17 0 00 000000         TX1:    RTN                             ; return

                                   ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

000031' 400000  000041'            TS1:    STABLE  (SSCHK,TS1CHK)          ; check if done yet
000032' 400000  000100'                    STABLE  (SSCHK,TS1LOD)          ; load ucode with special jump field
000033' 100100  000000                     STABLE  (SSSTRT,1,0,0)          ; do 1st JMAP
000034' 100100  010000                     STABLE  (SSSTRT,1,1,0)          ; do 2nd JMAP
000035' 100100  020000                     STABLE  (SSSTRT,1,2,0)          ; do 3rd JMAP
000036' 100100  030000                     STABLE  (SSSTRT,1,3,0)          ; do 4th JMAP
000037' 500000  000031'                    STABLE  (SSJRST,TS1)            ; loop till done
000040' 000000  000000                     STABLE  (SSLAST)

                                   ; Check if pattern table complete

000041' 350 00 0 00 000007         TS1CHK: AOS     7                       ; increment table pointer
000042' 200 01 0 07 000000                 MOVE    1,(7)                   ; get pattern - done?
000043' 202 01 0 00 012214'                MOVEM   1,DATPAT#               ; save data pattern
000044' 331 00 0 00 000001                 SKIPL   1                       ; done with patterns?
000045' 350 00 0 17 000000                 AOS     (P)                     ; no - return +1
000046' 263 17 0 00 000000                 RTN                             ; return
```

```

                                        ; Data patterns

000047' 000000  000000                  TS1PAT: 0
000050' 000000  005777                          5777
000051' 000000  000001                          0001
000052' 000000  000002                          0002
000053' 000000  000004                          0004
000054' 000000  000010                          0010
000055' 000000  000020                          0020
000056' 000000  000040                          0040
000057' 000000  000100                          0100
000060' 000000  000200                          0200
000061' 000000  000400                          0400
000062' 000000  001000                          1000
000063' 000000  002000                          2000
000064' 000000  004000                          4000
000065' 000000  005776                          5776
000066' 000000  005775                          5775
000067' 000000  005773                          5773
000070' 000000  005767                          5767
000071' 000000  005757                          5757
000072' 000000  005737                          5737
000073' 000000  005677                          5677
000074' 000000  005577                          5577
000075' 000000  005377                          5377
000076' 000000  003777                          3777
000077' 777777  777777                          -1

                                        ; Load microcode with special jump field / Set up end address

000100' 400 02 0 00 000000              TS1LOD: SETZ    2,                              ; init microword counter
000101' 200 00 0 02 000130'                     MOVE    T1M(2)                          ; get 1st half of 1st uword
000102' 404 00 0 00 010450'                     AND     [777700007777]                  ; mask out address given
000103' 200 01 0 00 012214'                     MOVE    1,DATPAT                        ; get proper address
000104' 137 01 0 00 010451'                     DPB     1,[POINT 12,0,23]               ; insert it
000105' 137 01 0 00 010452'                     DPB     1,[POINT 12,TS1+2,35]           ; set up end address
000106' 137 01 0 00 010453'                     DPB     1,[POINT 12,TS1+3,35]           ; set up end address
000107' 137 01 0 00 010454'                     DPB     1,[POINT 12,TS1+4,35]           ; set up end address
000110' 137 01 0 00 010455'                     DPB     1,[POINT 12,TS1+5,35]           ; set up end address
000111' 202 00 0 02 000130'                     MOVEM   T1M(2)                          ; update the microword
000112' 271 02 0 00 000002                      ADDI    2,2                             ; point to next one
000113' 305 02 0 00 000010                      CAIGE   2,^D8                           ; reach end yet?
000114' 254 00 0 00 000101'                     JRST    TS1LOD+1                        ; no - loop till done
000115' 316 16 0 00 000000*                     CAMN    MBCN,PORTNI                     ; NI port?
000116' 476 00 0 00 000000*                     SETOM   TSLOD1                          ; yes - must reload microcode each time
000117' 316 16 0 00 000000*                     CAMN    MBCN,PORTCI                     ; CI port?
000120' 476 00 0 00 000000*                     SETOM   TSLOD2                          ; yes - must reload microcode each time
000121' 201 01 0 00 000130'                     MOVEI   1,T1M                           ; set up microcode address
000122' 260 17 0 00 000000*                     GO      TLOAD                           ; load/verify it
000123' 334 00 0 00 000000                      SKIPA                                   ; failed - exit test
000124' 350 00 0 17 000000                      AOS     (P)                             ; set up ok return
000125' 263 17 0 00 000000                      RTN                                     ; yes - return
```

```
  186
  187                                        ; Error messages
  188
  189  000126' 140000  010456'              MA1:    MSG!TXNOT![ASCIZ /JMAP did not dispatch to correct location/]
  190  000127' 000000000000#                        LAST!CALL!TXALL!SSPNT          ; print sequencer data
  191
  192                                        ; Test microcode
  193
  194  000130' 000000  000000               T1M:    MWORD   <ADDR=0,JMAP,J=0,DISA,CCOF>     ; disa/off
  195  000131' 000000  010040
  196  000132' 000100  000000                       MWORD   <JMAP,J=0,DISA,CCON>            ; disa/on       (1)
  197  000133' 000000  030040
  198  000134' 000200  000000                       MWORD   <JMAP,J=0,ENA,CCOF>             ; ena/off       (2)
  199  000135' 000400  010040
  200  000136' 000300  000000                       MWORD   <JMAP,J=0,ENA,CCON>             ; ena/on        (3)
  201  000137' 000400  030040
  202  000140' 777777  777777                       -1
```

```

                                 ;#**********************************************************************
                                 ;* Test 2 - JZ Instruction
                                 ;
                                 ; Description:  Verify that the  JZ  instruction  unconditionally
                                 ;               causes  a  jump to  CRAM address 0.  Execute a JZ
                                 ;               instruction at location 5777  and verify the next
                                 ;               CRAM address goes to zero.  Repeat  with location
                                 ;               3777.
                                 ;
                                 ;               The JZ also resets the stack - that it does so is
                                 ;               tested in subsequent tests.
                                 ;
                                 ; Procedure:    Clear Port
                                 ;               Load test microcode
                                 ;
                                 ;               Load RAR with 1
                                 ;               Set MPRUN/SINGLE STEP (execute JMAP,JZ)
                                 ;               Read LAR, and verify address = 0
                                 ;
                                 ;               Load RAR with 2
                                 ;               Set MPRUN/SINGLE STEP (execute JMAP,JZ)
                                 ;               Read LAR, and verify address = 0
                                 ;
                                 ;               Load RAR with 3
                                 ;               Set MPRUN/SINGLE STEP (execute JMAP,JZ)
                                 ;               Read LAR, and verify address = 0
                                 ;
                                 ;               Load RAR with 4
                                 ;               Set MPRUN/SINGLE STEP (execute JMAP,JZ)
                                 ;               Read LAR, and verify address = 0
                                 ;
                                 ; Failure:      ---
                                 ;#**********************************************************************

                                 ; Test data

000141' 254 00 0 00 000152'      TSTS2:  JRST    TG2                             ; go start test
000142' 240401  000002                   SEQ!MPROC!NDMP!ZSEQ!2                   ; test mask
000143' 000203' 010467'                  T2M,,[ASCIZ ^JZ Instruction^]
000144' 010442' 010443'                  [EXP MLAST!M7],,[EXP M16,M9,E17,M1,MLAST!M8]
000145' 000000  000226'                  TSTS3                                   ; failure test table
000146' 000000  000327'                  TSTS4                                   ;   ...
000147' 000000  000430'                  TSTS5
000150' 000000  000564'                  TSTS6
000151' 777777  777777                   -1
```

```

                                     ; Start test

000152' 201 00 0 00 000000'          TG2:    MOVEI   Z3                      ; get address of module start
000153' 260 17 0 00 000012*                  GO      TRACE                   ; handle trace output
000154' 201 01 0 00 000203'                  MOVEI   1,T2M                   ; set up microcode address
000155' 260 17 0 00 000122*                  GO      TLOAD                   ; load/verify it
000156' 263 17 0 00 000000                   RTN                             ; failed - exit test

                                     ; Initialization

000157' 260 17 0 00 000013*          TL2:    GO      IPACLR                  ; clear port
000160' 402 00 0 00 000014*                  SETZM   TSTSUB                  ; initialize subtest number
000161' 201 06 0 00 000174'                  MOVEI   6,TS2                   ; get sstep table address

                                     ; Loop on single step table entries

000162' 400 15 0 00 000000           TA2:    SETZ    ERFLG,                  ; clear error flag
000163' 260 17 0 00 000020*                  GO      SEXEC                   ; execute table entry
000164' 254 00 0 00 000173'                  JRST    TX2                     ; end of sstep table
000165' 254 00 0 00 000162'                  JRST    TA2                     ; keep looping after call
000166' 474 15 0 00 000000                   SETO    ERFLG,                  ; error - end address incorrect

                                     ; Handle error printouts and scope looping

000167' 027 00 0 00 000201'                  SCOPER  MA2                     ; print error message
000170' 254 00 0 00 000157'                  JRST    TL2                     ; loop on error
000171' 254 00 0 00 000173'                  JRST    TX2                     ; altmode exit
000172' 322 15 0 00 000162'                  JUMPE   ERFLG,TA2               ; do next entry (if no error yet)

                                     ; End of test

000173' 263 17 0 00 000000           TX2:    RTN                             ; return

                                     ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

000174' 100200  010000               TS2:    STABLE  (SSSTRT,2,1,0)
000175' 100200  020000                       STABLE  (SSSTRT,2,2,0)
000176' 100200  030000                       STABLE  (SSSTRT,2,3,0)
000177' 100200  040000                       STABLE  (SSSTRT,2,4,0)
000200' 000000  000000                       STABLE  (SSLAST)

                                     ; Error messages

000201' 140000  010472'              MA2:    MSG!TXNOT![ASCIZ /JZ did not go to location 0/]
000202' 000000000000#                        LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```

                                    ; Test microcode

000203' 000000  000000              T2M:    MWORD   <ADDR=0,JZ>
000204' 000000  000000
000205' 000137  760000                      MWORD   <JMAP,J=3776>                   ; 1
000206' 000000  000040
000207' 000237  770000                      MWORD   <JMAP,J=3777>                   ; 2
000210' 000000  000040
000211' 000357  760000                      MWORD   <JMAP,J=5776>                   ; 3
000212' 000000  000040
000213' 000457  770000                      MWORD   <JMAP,J=5777>                   ; 4
000214' 000000  000040

000215' 377600  770000                      MWORD   <ADDR=3776,JZ,J=77,DISA,CCOF>   ; disa/off
000216' 000000  010000
000217' 377700  770000                      MWORD   <ADDR=3777,JZ,J=77,ENA,CCOF>    ; disa/off
000220' 000400  010000
000221' 577600  770000                      MWORD   <ADDR=5776,JZ,J=77,DISA,CCON>   ; disa/on
000222' 000000  030000
000223' 577700  770000                      MWORD   <ADDR=5777,JZ,J=77,ENA,CCON>    ; disa/on
000224' 000400  030000
000225' 777777  777777                      -1
```

```
                                ;#********************************************************************
                                ;* Test 3 - JZ Instruction - Effect on Reg/Ctr
                                ;
                                ; Description:  Verify that the JZ instruction  does  not  affect
                                ;               the  contents  of the register/counter.  Load the
                                ;               counter, do a JZ, and branch to register contents
                                ;               to see if it changed.
                                ;
                                ; Procedure:    Clear Port
                                ;               Load test microcode
                                ;
                                ;               Set RAR to 100 (Test En/Cond=OFF/OFF)
                                ;               Set MPRUN/SINGLE STEP (execute JMAP,LDCT,JZ,JRP)
                                ;               Read LAR, and verify address = 777
                                ;
                                ;               Set RAR to 103 (Test En/Cond=OFF/ON)
                                ;               Set MPRUN/SINGLE STEP (execute JMAP,LDCT,JZ,JRP)
                                ;               Read LAR, and verify address = 777
                                ;
                                ;               Set RAR to 106 (Test En/Cond=ON/OFF)
                                ;               Set MPRUN/SINGLE STEP (execute JMAP,LDCT,JZ,JRP)
                                ;               Read LAR, and verify address = 777
                                ;
                                ;               Set RAR to 111 (Test En/Cond=ON/ON)
                                ;               Set MPRUN/SINGLE STEP (execute JMAP,LDCT,JZ,JRP)
                                ;               Read LAR, and verify address = 777
                                ;
                                ; Failure:      ---
                                ;#********************************************************************

                                ; Test data

000226' 254 00 0 00 000237'     TSTS3:  JRST    TG3                     ; go start test
000227' 240401  000003                  SEQ!MPROC!NDMP!ZSEQ!3           ; test mask
000230' 000270' 010500'                 T3M,,[ASCIZ ^JZ Instruction - Effect on Reg/Ctr^]
000231' 010442' 010443'                 [EXP MLAST!M7],,[EXP M16,M9,E17,M1,MLAST!M8]
000232' 000000  000327'                 TSTS4                           ; failure test table
000233' 000000  000430'                 TSTS5                           ;   ...
000234' 000000  000564'                 TSTS6
000235' 000000  000637'                 TSTS7
000236' 777777  777777                  -1

                                ; Start test

000237' 201 00 0 00 000000'     TG3:    MOVEI   Z3                      ; get address of module start
000240' 260 17 0 00 000153*             GO      TRACE                   ; handle trace output
000241' 201 01 0 00 000270'             MOVEI   1,T3M                   ; set up microcode address
000242' 260 17 0 00 000155*             GO      TLOAD                   ; load/verify it
000243' 263 17 0 00 000000              RTN                             ; failed - exit test
```

```

                                        ; Initialization

000244' 260 17 0 00 000157*     TL3:    GO      IPACLR                          ; clear port
000245' 402 00 0 00 000160*             SETZM   TSTSUB                          ; initialize subtest number
000246' 201 06 0 00 000261'             MOVEI   6,TS3                           ; get sstep table address

                                        ; Loop on single step table entries

000247' 400 15 0 00 000000      TA3:    SETZ    ERFLG,                          ; clear error flag
000250' 260 17 0 00 000163*             GO      SEXEC                           ; execute table entry
000251' 254 00 0 00 000260'             JRST    TX3                             ; end of sstep table
000252' 254 00 0 00 000247'             JRST    TA3                             ; keep looping after call
000253' 474 15 0 00 000000              SETO    ERFLG,                          ; error - end address incorrect

                                        ; Handle error printouts and scope looping

000254' 027 00 0 00 000266'             SCOPER  MA3                             ; print error message
000255' 254 00 0 00 000244'             JRST    TL3                             ; loop on error
000256' 254 00 0 00 000260'             JRST    TX3                             ; altmode exit
000257' 322 15 0 00 000247'             JUMPE   ERFLG,TA3                       ; do next entry (if no error yet)

                                        ; End of test

000260' 263 17 0 00 000000      TX3:    RTN                                     ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

000261' 100401  000777          TS3:    STABLE  (SSSTRT,4,100,777)
000262' 100401  030777                  STABLE  (SSSTRT,4,103,777)
000263' 100401  060777                  STABLE  (SSSTRT,4,106,777)
000264' 100401  110777                  STABLE  (SSSTRT,4,111,777)
000265' 000000  000000                  STABLE  (SSLAST)

                                        ; Error messages

000266' 140000  010507'         MA3:    MSG!TXNOT![ASCIZ ^JZ affected Reg/Ctr^]
000267' 000000000000#                   LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```


                                                  ; Test microcode

000270' 000004  000000                            T3M:    MWORD   <ADDR=0,JRP,ENA,CCOF,J=400>     ; jump to cntr addr
000271' 000400  010160
000272' 010001  010000                                    MWORD   <ADDR=100,JMAP,J=101>           ;                 (100)
000273' 000000  000040
000274' 010107  770000                                    MWORD   <LDCT,J=777>                    ; load counter    (101)
000275' 000000  000300
000276' 010204  000000                                    MWORD   <JZ,J=400,DISA,CCOF>            ; disa/off        (102)
000277' 000000  010000
000300' 010301  040000                                    MWORD   <JMAP,J=104>                    ;                 (103)
000301' 000000  000040
000302' 010407  770000                                    MWORD   <LDCT,J=777>                    ; load counter    (104)
000303' 000000  000300
000304' 010504  000000                                    MWORD   <JZ,J=400,DISA,CCON>            ; disa/on         (105)
000305' 000000  030000
000306' 010601  070000                                    MWORD   <JMAP,J=107>                    ;                 (106)
000307' 000000  000040
000310' 010707  770000                                    MWORD   <LDCT,J=777>                    ; load counter    (107)
000311' 000000  000300
000312' 011004  000000                                    MWORD   <JZ,J=400,ENA,CCOF>             ; ena/off         (110)
000313' 000400  010000
000314' 011101  120000                                    MWORD   <JMAP,J=112>                    ;                 (111)
000315' 000000  000040
000316' 011207  770000                                    MWORD   <LDCT,J=777>                    ; load counter    (112)
000317' 000000  000300
000320' 011304  000000                                    MWORD   <JZ,J=400,ENA,CCON>             ; ena/on          (113)
000321' 000400  030000

000322' 040004  000000                                    MWORD   <ADDR=400,JMAP,J=400>           ; error if here   (400)
000323' 000000  000040
000324' 077704  000000                                    MWORD   <ADDR=777,JMAP,J=400>           ; succeed if here (777)
000325' 000000  000040
000326' 777777  777777                                    -1
```

```

                                    ;#*****************************************************************
                                    ;* Test 4 - JZ Instruction - Effect on Reg/Ctr
                                    ;
                                    ; Description:  Verify that the JZ instruction  does  not  affect
                                    ;               the contents of the register/counter.   Load  the
                                    ;               counter, do a JZ, and branch to register contents
                                    ;               to see if it changed.
                                    ;
                                    ; Procedure:    Clear Port
                                    ;               Load test microcode
                                    ;
                                    ;               Set RAR to 100 (Test En/Cond=OFF/OFF)
                                    ;               Set MPRUN/SINGLE STEP (execute JMAP,LDCT,JZ,JRP)
                                    ;               Read LAR, and verify address = 777
                                    ;
                                    ;               Set RAR to 103 (Test En/Cond=OFF/ON)
                                    ;               Set MPRUN/SINGLE STEP (execute JMAP,LDCT,JZ,JRP)
                                    ;               Read LAR, and verify address = 777
                                    ;
                                    ;               Set RAR to 106 (Test En/Cond=ON/OFF)
                                    ;               Set MPRUN/SINGLE STEP (execute JMAP,LDCT,JZ,JRP)
                                    ;               Read LAR, and verify address = 777
                                    ;
                                    ;               Set RAR to 111 (Test En/Cond=ON/ON)
                                    ;               Set MPRUN/SINGLE STEP (execute JMAP,LDCT,JZ,JRP)
                                    ;               Read LAR, and verify address = 777
                                    ;
                                    ; Failure:      ---
                                    ;#*****************************************************************

                                    ; Test data

 000327' 254 00 0 00 000340'        TSTS4:  JRST    TG4                             ; go start test
 000330' 240401  000004                     SEQ!MPROC!NDMP!ZSEQ!4                   ; test mask
 000331' 000371' 010500'                    T4M,,[ASCIZ ^JZ Instruction - Effect on Reg/Ctr^]
 000332' 010442' 010443'                    [EXP MLAST!M7],,[EXP M16,M9,E17,M1,MLAST!M8]
 000333' 000000  000430'                    TSTS5                                   ; failure test table
 000334' 000000  000564'                    TSTS6                                   ;   ...
 000335' 000000  000637'                    TSTS7
 000336' 000000  000712'                    TSTS10
 000337' 777777  777777                     -1

                                    ; Start test

 000340' 201 00 0 00 000000'        TG4:    MOVEI   Z3                              ; get address of module start
 000341' 260 17 0 00 000240*                GO      TRACE                           ; handle trace output
 000342' 201 01 0 00 000371'                MOVEI   1,T4M                           ; set up microcode address
 000343' 260 17 0 00 000242*                GO      TLOAD                           ; load/verify it
 000344' 263 17 0 00 000000                 RTN                                     ; failed - exit test
```

```
                                        ; Initialization

000345' 260 17 0 00 000244*             TL4:    GO      IPACLR                  ; clear port
000346' 402 00 0 00 000245*                     SETZM   TSTSUB                  ; initialize subtest number
000347' 201 06 0 00 000362'                     MOVEI   6,TS4                   ; get sstep table address

                                        ; Loop on single step table entries

000350' 400 15 0 00 000000              TA4:    SETZ    ERFLG,                  ; clear error flag
000351' 260 17 0 00 000250*                     GO      SEXEC                   ; execute table entry
000352' 254 00 0 00 000361'                     JRST    TX4                     ; end of sstep table
000353' 254 00 0 00 000350'                     JRST    TA4                     ; keep looping after call
000354' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - end address incorrect

                                        ; Handle error printouts and scope looping

000355' 027 00 0 00 000367'                     SCOPER  MA4                     ; print error message
000356' 254 00 0 00 000345'                     JRST    TL4                     ; loop on error
000357' 254 00 0 00 000361'                     JRST    TX4                     ; altmode exit
000360' 322 15 0 00 000350'                     JUMPE   ERFLG,TA4               ; do next sstep table entry

                                        ; End of test

000361' 263 17 0 00 000000              TX4:    RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

000362' 100401  000777                  TS4:    STABLE  (SSSTRT,4,100,777)
000363' 100401  030777                          STABLE  (SSSTRT,4,103,777)
000364' 100401  060777                          STABLE  (SSSTRT,4,106,777)
000365' 100401  110777                          STABLE  (SSSTRT,4,111,777)
000366' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

000367' 140000  010507'                 MA4:    MSG!TXNOT![ASCIZ ^JZ affected Reg/Ctr^]
000370' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```

                                        ; Test microcode

000371' 000004  000000                  T4M:    MWORD   <ADDR=0,JRP,J=400,ENA,CCOF>     ; jump to cntr addr
000372' 000400  010160
000373' 010001  010000                          MWORD   <ADDR=100,JMAP,J=101>           ;                  (100)
000374' 000000  000040
000375' 010107  770000                          MWORD   <LDCT,J=777>                    ; load counter     (101)
000376' 000000  000300
000377' 010200  000000                          MWORD   <JZ,J=0,DISA,CCOF>              ; disa/off         (102)
000400' 000000  010000
000401' 010301  040000                          MWORD   <JMAP,J=104>                    ;                  (103)
000402' 000000  000040
000403' 010407  770000                          MWORD   <LDCT,J=777>                    ; load counter     (104)
000404' 000000  000300
000405' 010500  000000                          MWORD   <JZ,J=0,DISA,CCON>              ; disa/on          (105)
000406' 000000  030000
000407' 010601  070000                          MWORD   <JMAP,J=107>                    ;                  (106)
000410' 000000  000040
000411' 010707  770000                          MWORD   <LDCT,J=777>                    ; load counter     (107)
000412' 000000  000300
000413' 011000  000000                          MWORD   <JZ,J=0,ENA,CCOF>               ; ena/off          (110)
000414' 000400  010000
000415' 011101  120000                          MWORD   <JMAP,J=112>                    ;                  (111)
000416' 000000  000040
000417' 011207  770000                          MWORD   <LDCT,J=777>                    ; load counter     (112)
000420' 000000  000300
000421' 011300  000000                          MWORD   <JZ,J=0,ENA,CCON>               ; ena/on           (113)
000422' 000400  030000

000423' 040004  000000                          MWORD   <ADDR=400,JMAP,J=400>           ; error if here    (400)
000424' 000000  000040
000425' 077704  000000                          MWORD   <ADDR=777,JMAP,J=400>           ; success if here  (777)
000426' 000000  000040
000427' 777777  777777                          -1
```

```

                                          ;#***********************************************************************
                                          ;* Test 5 - CONT Instruction
                                          ;
                                          ; Description:  Verify that the CONT instruction  unconditionally
                                          ;               increments CRAM address by one.  Load  CRAM  test
                                          ;               address  with  a  CONT, then verify that the CONT
                                          ;               instruction produces test address + 1.
                                          ;
                                          ; Procedure:    Clear Port
                                          ;               Load test microcode and set RAR to 0
                                          ;
                                          ;               Set MPRUN/SINGLE STEP (execute JMAP)
                                          ;               Set MPRUN/SINGLE STEP (execute CONT)
                                          ;               Read LAR, and verify address = 2
                                          ;               Set MPRUN/SINGLE STEP (execute CONT)
                                          ;               Read LAR, and verify address = 3
                                          ;               Set MPRUN/SINGLE STEP (execute CONT)
                                          ;               Read LAR, and verify address = 4
                                          ;               Set MPRUN/SINGLE STEP (execute CONT)
                                          ;               Read LAR, and verify address = 5
                                          ;
                                          ;               Set MPRUN/SINGLE STEP (execute JMAP,CONT)
                                          ;               Read LAR, and verify address = 10
                                          ;
                                          ;               Repeat above 2 steps for CONT resulting
                                          ;               in LAR addresses of:  10,20,40,100,400,
                                          ;               1000,2000,4000
                                          ;
                                          ; Failure:      ---
                                          ;#***********************************************************************

                                          ; Test data

000430' 254 00 0 00 000441'               TSTS5:  JRST    TG5                     ; go start test
000431' 240401  000005                            SEQ!MPROC!NDMP!ZSEQ!5           ; test mask
000432' 000503' 010513'                           T5M,,[ASCIZ ^CONT Instruction^]
000433' 010442' 010517'                           [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
000434' 000000  000564'                           TSTS6                           ; failure test table
000435' 000000  000637'                           TSTS7                           ;   ...
000436' 000000  000712'                           TSTS10
000437' 000000  000765'                           TSTS11
000440' 777777  777777                            -1

                                          ; Start test

000441' 201 00 0 00 000000'               TG5:    MOVEI   Z3                      ; get address of module start
000442' 260 17 0 00 000341*                       GO      TRACE                   ; handle trace output
000443' 201 01 0 00 000503'                       MOVEI   1,T5M                   ; set up microcode address
000444' 260 17 0 00 000343*                       GO      TLOAD                   ; load/verify it
000445' 263 17 0 00 000000                        RTN                             ; failed - exit test
```

```

                                        ; Initialization

000446' 260 17 0 00 000345*     TL5:    GO      IPACLR                  ; clear port
000447' 402 00 0 00 000346*             SETZM   TSTSUB                  ; initialize subtest number
000450' 201 06 0 00 000463'             MOVEI   6,TS5                   ; get sstep table address

                                        ; Loop on single step table entries

000451' 400 15 0 00 000000      TA5:    SETZ    ERFLG,                  ; clear error flag
000452' 260 17 0 00 000351*             GO      SEXEC                   ; execute table entry
000453' 254 00 0 00 000462'             JRST    TX5                     ; end of sstep table
000454' 254 00 0 00 000451'             JRST    TA5                     ; keep looping after call
000455' 474 15 0 00 000000              SETO    ERFLG,                  ; error - end address incorrect

                                        ; Handle error printouts and scope looping

000456' 027 00 0 00 000501'             SCOPER  MA5                     ; print error message
000457' 254 00 0 00 000446'             JRST    TL5                     ; loop on error
000460' 254 00 0 00 000462'             JRST    TX5                     ; altmode exit
000461' 322 15 0 00 000451'             JUMPE   ERFLG,TA5               ; do next sstep table entry

                                        ; End of test

000462' 263 17 0 00 000000      TX5:    RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

000463' 100200  000002          TS5:    STABLE  (SSSTRT,2,0,2)
000464' 200100  000003                  STABLE  (SSCONT,1,3)
000465' 200100  000004                  STABLE  (SSCONT,1,4)
000466' 200100  000005                  STABLE  (SSCONT,1,5)
000467' 100200  060010                  STABLE  (SSSTRT,2,6,10)
000470' 100200  160020                  STABLE  (SSSTRT,2,16,20)
000471' 100200  360040                  STABLE  (SSSTRT,2,36,40)
000472' 100200  760100                  STABLE  (SSSTRT,2,76,100)
000473' 100201  760200                  STABLE  (SSSTRT,2,176,200)
000474' 100203  760400                  STABLE  (SSSTRT,2,376,400)
000475' 100207  761000                  STABLE  (SSSTRT,2,776,1000)
000476' 100217  762000                  STABLE  (SSSTRT,2,1776,2000)
000477' 100237  764000                  STABLE  (SSSTRT,2,3776,4000)
000500' 000000  000000                  STABLE  (SSLAST)

                                        ; Error messages

000501' 140000  010523'         MA5:    MSG!TXNOT![ASCIZ /CONT did not go to PC+1/]
000502' 000000000000#                   LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```

                                        ; Test microcode

000503' 000000  010000          T5M:    MWORD   <ADDR=0,JMAP,J=1>       ; 0
000504' 000000  000040
000505' 000100  440000                  MWORD   <CONT,J=44>             ; 1
000506' 000000  000340
000507' 000200  440000                  MWORD   <CONT,J=44>             ; 2
000510' 000000  000340
000511' 000300  440000                  MWORD   <CONT,J=44>             ; 3
000512' 000000  000340
000513' 000400  440000                  MWORD   <CONT,J=44>             ; 4
000514' 000000  000340
000515' 000600  070000                  MWORD   <ADDR=6,JMAP,J=7>       ; 6
000516' 000000  000040
000517' 000700  440000                  MWORD   <CONT,J=44>             ; 7
000520' 000000  000340
000521' 001600  170000                  MWORD   <ADDR=16,JMAP,J=17>     ; 16
000522' 000000  000040
000523' 001700  440000                  MWORD   <CONT,J=44>             ; 17
000524' 000000  000340
000525' 003600  370000                  MWORD   <ADDR=36,JMAP,J=37>     ; 36
000526' 000000  000040
000527' 003700  440000                  MWORD   <CONT,J=44>             ; 37
000530' 000000  000340
000531' 007600  770000                  MWORD   <ADDR=76,JMAP,J=77>     ; 76
000532' 000000  000040
000533' 007700  440000                  MWORD   <CONT,J=44>             ; 77
000534' 000000  000340
000535' 017601  770000                  MWORD   <ADDR=176,JMAP,J=177>   ; 176
000536' 000000  000040
000537' 017700  440000                  MWORD   <CONT,J=44>             ; 177
000540' 000000  000340
000541' 037603  770000                  MWORD   <ADDR=376,JMAP,J=377>   ; 376
000542' 000000  000040
000543' 037700  440000                  MWORD   <CONT,J=44>             ; 377
000544' 000000  000340
000545' 077607  770000                  MWORD   <ADDR=776,JMAP,J=777>   ; 776
000546' 000000  000040
000547' 077700  440000                  MWORD   <CONT,J=44>             ; 777
000550' 000000  000340
000551' 177617  770000                  MWORD   <ADDR=1776,JMAP,J=1777> ; 1776
000552' 000000  000040
000553' 177700  440000                  MWORD   <CONT,J=44>             ; 1777
000554' 000000  000340
000555' 377637  770000                  MWORD   <ADDR=3776,JMAP,J=3777> ; 3776
000556' 000000  000040
000557' 377700  440000                  MWORD   <CONT,J=44>             ; 3777
000560' 000000  000340
000561' 004400  440000                  MWORD   <ADDR=44,JMAP,J=44>     ; error address
000562' 000000  000040
000563' 777777  777777                  -1
```

```
                                        ;#*******************************************************************
                                        ;* Test 6 - CJP Instruction
                                        ;
                                        ; Description:  Verify  that the CJP instruction causes a jump to
                                        ;               CRAM address  specified  by  the direct  inputs if
                                        ;               the TEST inputs are satisfied and does  not  jump
                                        ;               if the TEST inputs are not satisfied.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode and set RAR to 0
                                        ;               Set MPRUN/SINGLE STEP (execute CJP)
                                        ;               Read LAR, and verify address = 37
                                        ;               Set MPRUN/SINGLE STEP (execute CJP)
                                        ;               Read LAR, and verify address = 77
                                        ;               Set MPRUN/SINGLE STEP (execute CJP)
                                        ;               Read LAR, and verify address = 137
                                        ;               Set MPRUN/SINGLE STEP (execute CJP)
                                        ;               Read LAR, and verify address = 140
                                        ;
                                        ; Failure:      ---
                                        ;#*******************************************************************

                                        ; Test data

 000564' 254 00 0 00 000575'            TSTS6:  JRST    TG6                     ; go start test
 000565' 240401  000006                         SEQ!MPROC!NDMP!ZSEQ!6           ; test mask
 000566' 000626' 010530'                        T6M,,[ASCIZ ^CJP Instruction^]
 000567' 010442' 010517'                        [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
 000570' 000000  000637'                        TSTS7                           ; failure test table
 000571' 000000  000712'                        TSTS10                          ;   ...
 000572' 000000  000765'                        TSTS11
 000573' 000000  001033'                        TSTS12
 000574' 777777  777777                         -1

                                        ; Start test

 000575' 201 00 0 00 000000'            TG6:    MOVEI   Z3                      ; get address of module start
 000576' 260 17 0 00 000442*                    GO      TRACE                   ; handle trace output
 000577' 201 01 0 00 000626'                    MOVEI   1,T6M                   ; set up microcode address
 000600' 260 17 0 00 000444*                    GO      TLOAD                   ; load/verify it
 000601' 263 17 0 00 000000                     RTN                             ; failed - exit test

                                        ; Initialization

 000602' 260 17 0 00 000446*            TL6:    GO      IPACLR                  ; clear port
 000603' 402 00 0 00 000447*                    SETZM   TSTSUB                  ; initialize subtest number
 000604' 201 06 0 00 000617'                    MOVEI   6,TS6                   ; get sstep table address
```

```

                                          ; Loop on single step table entries

 000605' 400 15 0 00 000000               TA6:    SETZ    ERFLG,                  ; clear error flag
 000606' 260 17 0 00 000452*                      GO      SEXEC                   ; execute table entry
 000607' 254 00 0 00 000616'                      JRST    TX6                     ; end of sstep table
 000610' 254 00 0 00 000605'                      JRST    TA6                     ; keep looping after call
 000611' 474 15 0 00 000000                       SETO    ERFLG,                  ; error - end address incorrect

                                          ; Handle error printouts and scope looping

 000612' 027 00 0 00 000624'                      SCOPE   MA6                     ; print error message
 000613' 254 00 0 00 000602'                      JRST    TL6                     ; loop on error
 000614' 254 00 0 00 000616'                      JRST    TX6                     ; altmode exit
 000615' 322 15 0 00 000605'                      JUMPE   ERFLG,TA6               ; do next sstep table entry

                                          ; End of test

 000616' 263 17 0 00 000000               TX6:    RTN                             ; return

                                          ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

 000617' 100100  000037                   TS6:    STABLE  (SSSTRT,1,0,37)
 000620' 200100  000077                           STABLE  (SSCONT,1,77)
 000621' 200100  000137                           STABLE  (SSCONT,1,137)
 000622' 200100  000140                           STABLE  (SSCONT,1,140)
 000623' 000000  000000                           STABLE  (SSLAST)

                                          ; Error messages

 000624' 140000  010534'                  MA6:    MSG!TXNOT![ASCIZ /CJP did not dispatch correctly/]
 000625' 000000000000#                            LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                          ; Test microcode

 000626' 000000  370000                   T6M:    MWORD   <ADDR=0,CJP,DISA,CCOF,J=37>     ; disa/off      (0)
 000627' 000000  010060
 000630' 003700  770000                           MWORD   <ADDR=37,CJP,DISA,CCON,J=77>    ; disa/on       (37)
 000631' 000000  030060
 000632' 007701  370000                           MWORD   <ADDR=77,CJP,CENA,CCON,J=137>   ; ena/on        (77)
 000633' 000400  030060
 000634' 013707  770000                           MWORD   <ADDR=137,CJP,CENA,CCOF,J=777>  ; ena/off       (137)
 000635' 000400  010060
 000636' 777777  777777                           -1
```

```

                                 ;#******************************************************************
                                 ;* Test 7 - CJV Instruction
                                 ;
                                 ; Description:  Verify  that the CJV instruction causes a jump if
                                 ;               the  TEST  inputs are satisfied and does not jump
                                 ;               if not.
                                 ;
                                 ; Procedure:    Clear Port
                                 ;               Load test microcode and set RAR to 0
                                 ;               Set MPRUN/SINGLE STEP (execute CJV)
                                 ;               Read LAR, and verify address = 37
                                 ;               Set MPRUN/SINGLE STEP (execute CJV)
                                 ;               Read LAR, and verify address = 77
                                 ;               Set MPRUN/SINGLE STEP (execute CJV)
                                 ;               Read LAR, and verify address = 137
                                 ;               Set MPRUN/SINGLE STEP (execute CJV)
                                 ;               Read LAR, and verify address = 140
                                 ;
                                 ; Failure:      ---
                                 ;#******************************************************************

                                 ; Test data

000637' 254 00 0 00 000650'      TSTS7:  JRST    TG7                     ; go start test
000640' 240401  000007                   SEQ!MPROC!NDMP!ZSEQ!7           ; test mask
000641' 000701' 010543'                  T7M,,[ASCIZ ^CJV Instruction^]
000642' 010442' 010517'                  [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
000643' 000000  000712'                  TSTS10                          ; failure test table
000644' 000000  000765'                  TSTS11                          ;   ...
000645' 000000  001033'                  TSTS12
000646' 000000  001113'                  TSTS13
000647' 777777  777777                   -1

                                 ; Start test

000650' 201 00 0 00 000000'      TG7:    MOVEI   Z3                      ; get address of module start
000651' 260 17 0 00 000576*              GO      TRACE                   ; handle trace output
000652' 201 01 0 00 000701'              MOVEI   1,T7M                   ; set up microcode address
000653' 260 17 0 00 000600*              GO      TLOAD                   ; load/verify it
000654' 263 17 0 00 000000               RTN                             ; failed - exit test

                                 ; Initialization

000655' 260 17 0 00 000602*      TL7:    GO      IPACLR                  ; clear port
000656' 402 00 0 00 000603*              SETZM   TSTSUB                  ; initialize subtest number
000657' 201 06 0 00 000672'              MOVEI   6,TS7                   ; get sstep table address
```

```

                                    ; Loop on single step table entries

000660' 400 15 0 00 000000          TA7:    SETZ    ERFLG,                          ; clear error flag
000661' 260 17 0 00 000606*                 GO      SEXEC                           ; execute table entry
000662' 254 00 0 00 000671'                 JRST    TX7                             ; end of sstep table
000663' 254 00 0 00 000660'                 JRST    TA7                             ; keep looping after call
000664' 474 15 0 00 000000                  SETO    ERFLG,                          ; error - end address incorrect

                                    ; Handle error printouts and scope looping

000665' 027 00 0 00 000677'                 SCOPER  MA7                             ; print error message
000666' 254 00 0 00 000655'                 JRST    TL7                             ; loop on error
000667' 254 00 0 00 000671'                 JRST    TX7                             ; altmode exit
000670' 322 15 0 00 000660'                 JUMPE   ERFLG,TA7                       ; do next sstep table entry

                                    ; End of test

000671' 263 17 0 00 000000          TX7:    RTN                                     ; return

                                    ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

000672' 100100  000037              TS7:    STABLE  (SSSTRT,1,0,37)
000673' 200100  000077                      STABLE  (SSCONT,1,77)
000674' 200100  000137                      STABLE  (SSCONT,1,137)
000675' 200100  000140                      STABLE  (SSCONT,1,140)
000676' 000000  000000                      STABLE  (SSLAST)

                                    ; Error messages

000677' 140000  010547'             MA7:    MSG!TXNOT![ASCIZ /CJV did not dispatch correctly/]
000700' 000000000000#                       LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                    ; Test microcode

000701' 000000  370000              T7M:    MWORD   <ADDR=0,CJV,DISA,CCOF,J=37>      ; disa/off      (0)
000702' 000000  010140
000703' 003700  770000                      MWORD   <ADDR=37,CJV,DISA,CCON,J=77>     ; disa/on       (37)
000704' 000000  030140
000705' 007701  370000                      MWORD   <ADDR=77,CJV,CENA,CCON,J=137>    ; ena/on        (77)
000706' 000400  030140
000707' 013707  770000                      MWORD   <ADDR=137,CJV,CENA,CCOF,J=777>   ; ena/off       (137)
000710' 000400  010140
000711' 777777  777777                      -1
```

```

                                           ;#**********************************************************************
                                           ;* Test 10 - CJS Instruction
                                           ;
                                           ; Description:    Verify that the  CJS  instruction  jumps  to  the
                                           ;                 specified subroutine address if the TEST input is
                                           ;                 satisfied, and does not if not.
                                           ;
                                           ; Procedure:      Clear Port
                                           ;                 Load test microcode and set RAR to 0
                                           ;                 Set MPRUN/SINGLE STEP (execute CJS)
                                           ;                 Read LAR, and verify address = 37
                                           ;                 Set MPRUN/SINGLE STEP (execute CJS)
                                           ;                 Read LAR, and verify address = 77
                                           ;                 Set MPRUN/SINGLE STEP (execute CJS)
                                           ;                 Read LAR, and verify address = 137
                                           ;                 Set MPRUN/SINGLE STEP (execute CJS)
                                           ;                 Read LAR, and verify address = 140
                                           ;
                                           ; Failure:        ---
                                           ;#**********************************************************************

                                           ; Test data

 000712' 254 00 0 00 000723'       TSTS10: JRST    TG10                        ; go start test
 000713' 240401  000010                    SEQ!MPROC!NDMP!ZSEQ!10              ; test mask
 000714' 000754' 010556'                   T10M,,[ASCIZ ^CJS Instruction^]
 000715' 010442' 010517'                   [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
 000716' 000000  000765'                   TSTS11                              ; failure test table
 000717' 000000  001033'                   TSTS12                              ;   ...
 000720' 000000  001113'                   TSTS13
 000721' 000000  001163'                   TSTS14
 000722' 777777  777777                    -1

                                           ; Start test

 000723' 201 00 0 00 000000'       TG10:   MOVEI   Z3                          ; get address of module start
 000724' 260 17 0 00 000651*               GO      TRACE                       ; handle trace output
 000725' 201 01 0 00 000754'               MOVEI   1,T10M                      ; set up microcode address
 000726' 260 17 0 00 000653*               GO      TLOAD                       ; load/verify it
 000727' 263 17 0 00 000000                RTN                                 ; failed - exit test

                                           ; Initialization

 000730' 260 17 0 00 000655*       TL10:   GO      IPACLR                      ; clear port
 000731' 402 00 0 00 000656*               SETZM   TSTSUB                      ; initialize subtest number
 000732' 201 06 0 00 000745'               MOVEI   6,TS10                      ; get sstep table address
```

```

                                        ; Loop on single step table entries

000733' 400 15 0 00 000000      TA10:   SETZ    ERFLG,                  ; clear error flag
000734' 260 17 0 00 000661*             GO      SEXEC                   ; execute table entry
000735' 254 00 0 00 000744'             JRST    TX10                    ; end of sstep table
000736' 254 00 0 00 000733'             JRST    TA10                    ; keep looping after call
000737' 474 15 0 00 000000              SETO    ERFLG,                  ; error - end address incorrect

                                ; Handle error printouts and scope looping

000740' 027 00 0 00 000752'             SCOPER  MA10                    ; print error message
000741' 254 00 0 00 000730'             JRST    TL10                    ; loop on error
000742' 254 00 0 00 000744'             JRST    TX10                    ; altmode exit
000743' 322 15 0 00 000733'             JUMPE   ERFLG,TA10              ; do next sstep table entry

                                ; End of test

000744' 263 17 0 00 000000      TX10:   RTN                             ; return

                                ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

000745' 100100  000037          TS10:   STABLE  (SSSTRT,1,0,37)
000746' 200100  000077                  STABLE  (SSCONT,1,77)
000747' 200100  000137                  STABLE  (SSCONT,1,137)
000750' 200100  000140                  STABLE  (SSCONT,1,140)
000751' 000000  000000                  STABLE  (SSLAST)

                                ; Error messages

000752' 140000  010562'         MA10:   MSG!TXNOT![ASCIZ /CJS did not dispatch correctly/]
000753' 000000000000#                   LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                ; Test microcode

000754' 000000  370000          T10M:   MWORD   <ADDR=0,CJS,DISA,CCOF,J=37>     ; disa/off      (0)
000755' 000000  010020
000756' 003700  770000                  MWORD   <ADDR=37,CJS,DISA,CCON,J=77>    ; disa/on       (37)
000757' 000000  030020
000760' 007701  370000                  MWORD   <ADDR=77,CJS,CENA,CCON,J=137>   ; ena/on        (77)
000761' 000400  030020
000762' 013707  770000                  MWORD   <ADDR=137,CJS,CENA,CCOF,J=777>  ; ena/off       (137)
000763' 000400  010020
000764' 777777  777777                  -1
```

```
                                   ;#*******************************************************************
                                   ;* Test 11 - CJS Instruction
                                   ;
                                   ; Description:  Verify  that the CJS instruction pushes 'adr + 1'
                                   ;               onto stack if TEST is asserted and that the CRTN
                                   ;               instruction  correctly  pops  it  off if TEST is
                                   ;               asserted. This test implicitly verifies that the
                                   ;               the  first stack location can hold the value '1'
                                   ;               and that  the  stack pointer works to the extent
                                   ;               of handling a single push/pop.
                                   ;
                                   ; Procedure:    Clear Port
                                   ;               Load test microcode and set RAR to 1
                                   ;               Set MPRUN/SINGLE STEP (execute the JZ)
                                   ;               Set MPRUN/SINGLE STEP (execute the CJS)
                                   ;               Set MPRUN/SINGLE STEP (execute the CRTN)
                                   ;               Read LAR, and verify address = 1
                                   ;
                                   ; Failure:      ---
                                   ;#*******************************************************************

                                   ; Test data

000765' 254 00 0 00 000776'        TSTS11: JRST    TG11                    ; go start test
000766' 240401  000011                     SEQ!MPROC!NDMP!ZSEQ!11          ; test mask
000767' 001024' 010556'                    T11M,,[ASCIZ ^CJS Instruction^]
000770' 010442' 010517'                    [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
000771' 000000  001033'                    TSTS12                          ; failure test table
000772' 000000  001113'                    TSTS13                          ;   ...
000773' 000000  001163'                    TSTS14
000774' 000000  001267'                    TSTS15
000775' 777777  777777                     -1

                                   ; Start test

000776' 201 00 0 00 000000'        TG11:   MOVEI   Z3                      ; get address of module start
000777' 260 17 0 00 000724*                GO      TRACE                   ; handle trace output
001000' 201 01 0 00 001024'                MOVEI   1,T11M                  ; set up microcode address
001001' 260 17 0 00 000726*                GO      TLOAD                   ; load/verify it
001002' 263 17 0 00 000000                 RTN                             ; failed - exit test

                                   ; Initialization

001003' 260 17 0 00 000730*        TL11:   GO      IPACLR                  ; clear port
001004' 402 00 0 00 000731*                SETZM   TSTSUB                  ; initialize subtest number
001005' 201 06 0 00 001020'                MOVEI   6,TS11                  ; get sstep table address
```

```

                                ; Loop on single step table entries

001006' 400 15 0 00 000000      TA11:   SETZ    ERFLG,                  ; clear error flag
001007' 260 17 0 00 000734*             GO      SEXEC                   ; execute table entry
001010' 254 00 0 00 001017'             JRST    TX11                    ; end of sstep table
001011' 254 00 0 00 001006'             JRST    TA11                    ; keep looping after call
001012' 474 15 0 00 000000              SETO    ERFLG,                  ; error - end address incorrect

                                ; Handle error printouts and scope looping

001013' 027 00 0 00 001022'             SCOPER  MA11                    ; print error message
001014' 254 00 0 00 001003'             JRST    TL11                    ; loop on error
001015' 254 00 0 00 001017'             JRST    TX11                    ; altmode exit
001016' 322 15 0 00 001006'             JUMPE   ERFLG,TA11              ; do next sstep table entry

                                ; End of test

001017' 263 17 0 00 000000      TX11:   RTN                             ; return

                                ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

001020' 100300  010001          TS11:   STABLE  (SSSTRT,3,1,1)
001021' 000000  000000                  STABLE  (SSLAST)

                                ; Error messages

001022' 140000  010571'         MA11:   MSG!TXNOT![ASCIZ ^CJS/CRTN did not push/pop stack correctly^]
001023' 000000000000#                   LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                ; Test microcode

001024' 000000  770000          T11M:   MWORD   <ADDR=0,CJS,J=77>               ; call subroutine       (0)
001025' 000000  000020
001026' 000100  000000                  MWORD   <ADDR=1,JZ>                     ; initialize stack      (1)
001027' 000000  000000
001030' 007700  000000                  MWORD   <ADDR=77,CRTN>                  ; return to 1           (77)
001031' 000000  000240
001032' 777777  777777                  -1
```

```

                                          ;#**************************************************************
                                          ;* Test 12 - CJS Instruction
                                          ;
                                          ; Description:  Verify that the CJS  instruction  does  not  push
                                          ;               'addr + 1' onto the stack if TEST is negated.
                                          ;
                                          ; Procedure:    Clear Port
                                          ;               Load test microcode and set RAR to 4
                                          ;               Set MPRUN/SINGLE STEP (execute the JZ)
                                          ;               Set MPRUN/SINGLE STEP (execute the CONT,CONT,CONT)
                                          ;               Set MPRUN/SINGLE STEP (execute the CJS,CJS)
                                          ;               Set MPRUN/SINGLE STEP (execute the CRTN)
                                          ;               Read LAR, and verify address = 4
                                          ;
                                          ; Failure:      ---
                                          ;#**************************************************************

                                          ; Test data

 001033' 254 00 0 00 001044'              TSTS12: JRST    TG12                    ; go start test
 001034' 240401  000012                           SEQ!MPROC!NDMP!ZSEQ!12          ; test mask
 001035' 001072' 010556'                          T12M,,[ASCIZ ^CJS Instruction^]
 001036' 010442' 010517'                          [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
 001037' 000000  001113'                          TSTS13                          ; failure test table
 001040' 000000  001163'                          TSTS14                          ;   ...
 001041' 000000  001267'                          TSTS15
 001042' 000000  001461'                          TSTS16
 001043' 777777  777777                           -1

                                          ; Start test

 001044' 201 00 0 00 000000'              TG12:   MOVEI   Z3                      ; get address of module start
 001045' 260 17 0 00 000777*                      GO      TRACE                   ; handle trace output
 001046' 201 01 0 00 001072'                      MOVEI   1,T12M                  ; set up microcode address
 001047' 260 17 0 00 001001*                      GO      TLOAD                   ; load/verify it
 001050' 263 17 0 00 000000                       RTN                             ; failed - exit test

                                          ; Initialization

 001051' 260 17 0 00 001003*              TL12:   GO      IPACLR                  ; clear port
 001052' 402 00 0 00 001004*                      SETZM   TSTSUB                  ; initialize subtest number
 001053' 201 06 0 00 001066'                      MOVEI   6,TS12                  ; get sstep table address

                                          ; Loop on single step table entries

 001054' 400 15 0 00 000000               TA12:   SETZ    ERFLG,                  ; clear error flag
 001055' 260 17 0 00 001007*                      GO      SEXEC                   ; execute table entry
 001056' 254 00 0 00 001065'                      JRST    TX12                    ; end of sstep table
 001057' 254 00 0 00 001054'                      JRST    TA12                    ; keep looping after call
 001060' 474 15 0 00 000000                       SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

001061' 027 00 0 00 001070'                     SCOPER  MA12                    ; print error message
001062' 254 00 0 00 001051'                     JRST    TL12                    ; loop on error
001063' 254 00 0 00 001065'                     JRST    TX12                    ; altmode exit
001064' 322 15 0 00 001054'                     JUMPE   ERFLG,TA12              ; do next sstep table entry

                                        ; End of test

001065' 263 17 0 00 000000              TX12:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

001066' 100700  040004                  TS12:   STABLE  (SSSTRT,7,4,4)
001067' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

001070' 140000  010602'                 MA12:   MSG!TXNOT![ASCIZ /CJS without test conditions met shouldn't have affected stack/]
001071' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

001072' 000000  000000                  T12M:   MWORD   <ADDR=0,CONT>                   ;                       (0)
001073' 000000  000340
001074' 000100  000000                          MWORD   <CONT>                          ;                       (1)
001075' 000000  000340
001076' 000200  000000                          MWORD   <CONT>                          ;                       (2)
001077' 000000  000340
001100' 000300  050000                          MWORD   <CJS,J=5>                       ; TEST satisfied        (3)
001101' 000000  000020
001102' 000400  000000                          MWORD   <JZ>                            ; initialize stack      (4)
001103' 000000  000000
001104' 000500  770000                          MWORD   <CJS,ENA,CCOF,J=77>             ; no stack push         (5)
001105' 000400  010020
001106' 000600  000000                          MWORD   <CRTN>                          ; return to 4           (6)
001107' 000000  000240

001110' 007700  770000                          MWORD   <ADDR=77,JMAP,J=77>             ; error if here         (77)
001111' 000000  000040
001112' 777777  777777                          -1
```

```

                                          ;#***********************************************************************
                                          ;* Test 13 - CRTN Instruction
                                          ;
                                          ; Description:  Verify that the conditional CRTN instruction does
                                          ;               not do a return if TEST inputs are not satisfied.
                                          ;
                                          ; Procedure:    Clear Port
                                          ;               Load test microcode and set RAR to 2
                                          ;               Set MPRUN/SINGLE STEP (execute JZ,CJS)
                                          ;               Set MPRUN/SINGLE STEP (execute CRTN)
                                          ;               Read LAR, and verify address = 100
                                          ;
                                          ; Failure:      ---
                                          ;#***********************************************************************

                                          ; Test data

 001113' 254 00 0 00 001124'              TSTS13: JRST    TG13                    ; go start test
 001114' 240401  000013                           SEQ!MPROC!NDMP!ZSEQ!13          ; test mask
 001115' 001152' 010617'                          T13M,,[ASCIZ ^CRTN Instruction^]
 001116' 010442' 010517'                          [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
 001117' 000000  001163'                          TSTS14                          ; failure test table
 001120' 000000  001267'                          TSTS15                          ;   ...
 001121' 000000  001461'                          TSTS16
 001122' 000000  001653'                          TSTS17
 001123' 777777  777777                           -1

                                          ; Start test

 001124' 201 00 0 00 000000'              TG13:   MOVEI   Z3                      ; get address of module start
 001125' 260 17 0 00 001045*                      GO      TRACE                   ; handle trace output
 001126' 201 01 0 00 001152'                      MOVEI   1,T13M                  ; set up microcode address
 001127' 260 17 0 00 001047*                      GO      TLOAD                   ; load/verify it
 001130' 263 17 0 00 000000                       RTN                             ; failed - exit test

                                          ; Initialization

 001131' 260 17 0 00 001051*              TL13:   GO      IPACLR                  ; clear port
 001132' 402 00 0 00 001052*                      SETZM   TSTSUB                  ; initialize subtest number
 001133' 201 06 0 00 001146'                      MOVEI   6,TS13                  ; get sstep table address

                                          ; Loop on single step table entries

 001134' 400 15 0 00 000000               TA13:   SETZ    ERFLG,                  ; clear error flag
 001135' 260 17 0 00 001055*                      GO      SEXEC                   ; execute table entry
 001136' 254 00 0 00 001145'                      JRST    TX13                    ; end of sstep table
 001137' 254 00 0 00 001134'                      JRST    TA13                    ; keep looping after call
 001140' 474 15 0 00 000000                       SETO    ERFLG,                  ; error - end address incorrect
```

```
                                        ; Handle error printouts and scope looping

001141' 027 00 0 00 001150'                     SCOPER  MA13                    ; print error message
001142' 254 00 0 00 001131'                     JRST    TL13                    ; loop on error
001143' 254 00 0 00 001145'                     JRST    TX13                    ; altmode exit
001144' 322 15 0 00 001134'                     JUMPE   ERFLG,TA13              ; do next sstep table entry

                                        ; End of test

001145' 263 17 0 00 000000              TX13:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

001146' 100300  020100                  TS13:   STABLE  (SSSTRT,3,2,100)
001147' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

001150' 140000  010623'                 MA13:   MSG!TXNOT![ASCIZ /CRTN did a return without test conditions satisfied/]
001151' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

001152' 000000  770000                  T13M:   MWORD   <ADDR=0,CJS,J=77>               ; call subroutine       (0)
001153' 000000  000020
001154' 000100  010000                          MWORD   <JMAP,J=1>                      ; error if here         (1)
001155' 000000  000040
001156' 000200  000000                          MWORD   <JZ>                            ; init stack pointer    (2)
001157' 000000  000000
001160' 000700  000000                          MWORD   <ADDR=77,CRTN,ENA,CCOF>         ; do not return         (77)
001161' 000400  010240
001162' 777777  777777                          -1
```

```

                                                ;#*******************************************************************
                                                ;* Test 14 - CRTN Instruction
                                                ;
                                                ; Description:  Verify that the conditional CRTN instruction does
                                                ;               a return if TEST inputs are asserted, and this is
                                                ;               not affected by the register/counter contents.
                                                ;
                                                ; Procedure:    Clear Port
                                                ;               Load test microcode and set RAR to 15
                                                ;               Set MPRUN/SINGLE STEP (execute JZ,LDCT,CJS,CRTN)
                                                ;               Set MPRUN/SINGLE STEP (execute LDCT,CJS,CRTN)
                                                ;               Set MPRUN/SINGLE STEP (execute LDCT,CJS,CRTN)
                                                ;               Set MPRUN/SINGLE STEP (execute LDCT,CJS,CRTN)
                                                ;               Set MPRUN/SINGLE STEP (execute LDCT,CJS,CRTN)
                                                ;               Set MPRUN/SINGLE STEP (execute LDCT,CJS,CRTN)
                                                ;               Read LAR, and verify address = 4
                                                ;
                                                ; Failure:      ---
                                                ;#*******************************************************************

                                                ; Test data

        001163' 254 00 0 00 001174'             TSTS14: JRST    TG14                    ; go start test
        001164' 240401  000014                          SEQ!MPROC!NDMP!ZSEQ!14          ; test mask
        001165' 001222' 010617'                         T14M,,[ASCIZ ^CRTN Instruction^]
        001166' 010442' 010517'                         [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
        001167' 000000  001267'                         TSTS15                          ; failure test table
        001170' 000000  001461'                         TSTS16                          ;   ...
        001171' 000000  001653'                         TSTS17
        001172' 000000  002047'                         TSTS20
        001173' 777777  777777                          -1

                                                ; Start test

        001174' 201 00 0 00 000000'             TG14:   MOVEI   Z3                      ; get address of module start
        001175' 260 17 0 00 001125*                     GO      TRACE                   ; handle trace output
        001176' 201 01 0 00 001222'                     MOVEI   1,T14M                  ; set up microcode address
        001177' 260 17 0 00 001127*                     GO      TLOAD                   ; load/verify it
        001200' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                                ; Initialization

        001201' 260 17 0 00 001131*             TL14:   GO      IPACLR                  ; clear port
        001202' 402 00 0 00 001132*                     SETZM   TSTSUB                  ; initialize subtest number
        001203' 201 06 0 00 001216'                     MOVEI   6,TS14                  ; get sstep table address
```

```

                                          ; Loop on single step table entries

   001204' 400 15 0 00 000000             TA14:   SETZ    ERFLG,                  ; clear error flag
   001205' 260 17 0 00 001135*                    GO      SEXEC                   ; execute table entry
   001206' 254 00 0 00 001215'                    JRST    TX14                    ; end of sstep table
   001207' 254 00 0 00 001204'                    JRST    TA14                    ; keep looping after call
   001210' 474 15 0 00 000000                     SETO    ERFLG,                  ; error - end address incorrect

                                          ; Handle error printouts and scope looping

   001211' 027 00 0 00 001220'                    SCOPER  MA14                    ; print error message
   001212' 254 00 0 00 001201'                    JRST    TL14                    ; loop on error
   001213' 254 00 0 00 001215'                    JRST    TX14                    ; altmode exit
   001214' 322 15 0 00 001204'                    JUMPE   ERFLG,TA14              ; do next sstep table entry

                                          ; End of test

   001215' 263 17 0 00 000000             TX14:   RTN                             ; return

                                          ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

   001216' 102300  150014                 TS14:   STABLE  (SSSTRT,^D19,15,14)
   001217' 000000  000000                         STABLE  (SSLAST)

                                          ; Error messages

   001220' 140000  010636'                MA14:   MSG!TXNOT![ASCIZ /CRTN either did not return or affected cntr/]
   001221' 000000000000#                          LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```


                                    ; Test microcode
001222' 000000  000000              T14M:  MWORD   <ADDR=0,LDCT,J=0>              ; zero counter         (0)
001223' 000000  000300
001224' 000100  770000                     MWORD   <CJS,J=77>                     ; call subroutine      (1)
001225' 000000  000020
001226' 000207  770000                     MWORD   <LDCT,J=777>                   ; set counter to 777   (2)
001227' 000000  000300
001230' 000300  770000                     MWORD   <CJS,J=77>                     ; call subroutine      (3)
001231' 000000  000020
001232' 000400  000000                     MWORD   <LDCT,J=0>                     ; zero counter         (4)
001233' 000000  000300
001234' 000501  000000                     MWORD   <CJS,J=100>                    ; call subroutine      (5)
001235' 000000  000020
001236' 000607  770000                     MWORD   <LDCT,J=777>                   ; set counter to 777   (6)
001237' 000000  000300
001240' 000701  000000                     MWORD   <CJS,J=100>                    ; call subroutine      (7)
001241' 000000  000020
001242' 001000  000000                     MWORD   <LDCT,J=0>                     ; zero counter         (10)
001243' 000000  000300
001244' 001101  010000                     MWORD   <CJS,J=101>                    ; call subroutine      (11)
001245' 000000  000020
001246' 001207  770000                     MWORD   <LDCT,J=777>                   ; set counter to 777   (12)
001247' 000000  000300
001250' 001301  010000                     MWORD   <CJS,J=101>                    ; call subroutine      (13)
001251' 000000  000020
001252' 001407  770000                     MWORD   <JMAP,J=777>                   ; done                 (14)
001253' 000000  000040
001254' 001500  000000                     MWORD   <JZ>                           ; initialize stack     (15)
001255' 000000  000000

001256' 007700  000000                     MWORD   <ADDR=77,CRTN,DISA,CCOF>       ; disa/off      (77)
001257' 000000  010240
001260' 010000  000000                     MWORD   <CRTN,DISA,CCON>               ; disa/on       (100)
001261' 000000  030240
001262' 010100  000000                     MWORD   <CRTN,ENA,CCON>                ; ena/on        (101)
001263' 000400  030240

001264' 077707  770000                     MWORD   <ADDR=777,JMAP,J=777>          ; error if here (777)
001265' 000000  000040
001266' 777777  777777                     -1
```

```

                                       ;#*********************************************************************
                                       ;* Test 15 - Stack location #1
                                       ;
                                       ; Description:  Verify that the first  stack  location  can  hold
                                       ;               various data patterns.
                                       ;
                                       ; Procedure:    Clear Port
                                       ;               Load test microcode and set RAR to 73
                                       ;               Set MPRUN/SINGLE STEP (execute JZ)
                                       ;               Set MPRUN/SINGLE STEP 66 times
                                       ;               Read LAR, and verify address = 5777
                                       ;
                                       ; Failure:      ---
                                       ;#*********************************************************************

                                       ; Test data

001267' 254 00 0 00 001300'            TSTS15: JRST    TG15                    ; go start test
001270' 240401  000015                         SEQ!MPROC!NDMP!ZSEQ!15          ; test mask
001271' 001326' 010647'                        T15M,,[ASCIZ ^Stack location #1^]
001272' 010442' 010517'                        [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
001273' 000000  001461'                        TSTS16                          ; failure test table
001274' 000000  001653'                        TSTS17                          ;   ...
001275' 000000  002047'                        TSTS20
001276' 000000  002237'                        TSTS21
001277' 777777  777777                         -1

                                       ; Start test

001300' 201 00 0 00 000000'            TG15:   MOVEI   Z3                      ; get address of module start
001301' 260 17 0 00 001175*                    GO      TRACE                   ; handle trace output
001302' 201 01 0 00 001326'                    MOVEI   1,T15M                  ; set up microcode address
001303' 260 17 0 00 001177*                    GO      TLOAD                   ; load/verify it
001304' 263 17 0 00 000000                     RTN                             ; failed - exit test

                                       ; Initialization

001305' 260 17 0 00 001201*            TL15:   GO      IPACLR                  ; clear port
001306' 402 00 0 00 001202*                    SETZM   TSTSUB                  ; initialize subtest number
001307' 201 06 0 00 001322'                    MOVEI   6,TS15                  ; get sstep table address

                                       ; Loop on single step table entries

001310' 400 15 0 00 000000             TA15:   SETZ    ERFLG,                  ; clear error flag
001311' 260 17 0 00 001205*                    GO      SEXEC                   ; execute table entry
001312' 254 00 0 00 001321'                    JRST    TX15                    ; end of sstep table
001313' 254 00 0 00 001310'                    JRST    TA15                    ; keep looping after call
001314' 474 15 0 00 000000                     SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

001315' 027 00 0 00 001324'                     SCOPER  MA15                    ; print error message
001316' 254 00 0 00 001305'                     JRST    TL15                    ; loop on error
001317' 254 00 0 00 001321'                     JRST    TX15                    ; altmode exit
001320' 322 15 0 00 001310'                     JUMPE   ERFLG,TA15              ; do next sstep table entry

                                        ; End of test

001321' 263 17 0 00 000000              TX15:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

001322' 110200  735777                  TS15:   STABLE  (SSSTRT,^D66,73,5777)
001323' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

001324' 140000  010653'                 MA15:   MSG!TXNOT![ASCIZ /Stack location #1 broken/]
001325' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

001326' 000000  740000                  T15M:   MWORD   <ADDR=0,CJS,J=74>               ; call subroutine       (0)
001327' 000000  000020
001330' 000100  740000                          MWORD   <CJS,J=74>                      ; call subroutine       (1)
001331' 000000  000020
001332' 000200  740000                          MWORD   <CJS,J=74>                      ; call subroutine       (2)
001333' 000000  000020
001334' 000300  740000                          MWORD   <CJS,J=74>                      ; call subroutine       (3)
001335' 000000  000020
001336' 000400  070000                          MWORD   <JMAP,J=7>                      ; go to next address    (4)
001337' 000000  000040
001340' 000700  740000                          MWORD   <ADDR=7,CJS,J=74>               ; call subroutine       (7)
001341' 000000  000020
001342' 001000  170000                          MWORD   <ADDR=10,JMAP,J=17>             ; go to next address    (10)
001343' 000000  000040
001344' 001700  740000                          MWORD   <ADDR=17,CJS,J=74>              ; call subroutine       (17)
001345' 000000  000020
001346' 002000  370000                          MWORD   <ADDR=20,JMAP,J=37>             ; go to next address    (20)
001347' 000000  000040
001350' 003700  740000                          MWORD   <ADDR=37,CJS,J=74>              ; call subroutine       (37)
001351' 000000  000020
001352' 004000  770000                          MWORD   <ADDR=40,JMAP,J=77>             ; go to next address    (40)
001353' 000000  000040
001354' 007700  740000                          MWORD   <ADDR=77,CJS,J=74>              ; call subroutine       (77)
001355' 000000  000020
001356' 010001  770000                          MWORD   <ADDR=100,JMAP,J=177>           ; go to next address    (100)
001357' 000000  000040
001360' 017700  740000                          MWORD   <ADDR=177,CJS,J=74>             ; call subroutine       (177)
001361' 000000  000020
001362' 020003  770000                          MWORD   <ADDR=200,JMAP,J=377>           ; go to next address    (200)
001363' 000000  000040
001364' 037700  740000                          MWORD   <ADDR=377,CJS,J=74>             ; call subroutine       (377)
```

```
  1470  001365' 000000  000020
  1471  001366' 040007  770000          MWORD   <ADDR=400,JMAP,J=777>           ; go to next address    (400)
  1472  001367' 000000  000040
  1473  001370' 077700  740000          MWORD   <ADDR=777,CJS,J=74>             ; call subroutine       (777)
  1474  001371' 000000  000020
  1475  001372' 100017  770000          MWORD   <ADDR=1000,JMAP,J=1777>         ; go to next address    (1000)
  1476  001373' 000000  000040
  1477  001374' 177700  740000          MWORD   <ADDR=1777,CJS,J=74>            ; call subroutine       (1777)
  1478  001375' 000000  000020
  1479  001376' 200037  770000          MWORD   <ADDR=2000,JMAP,J=3777>         ; go to next address    (2000)
  1480  001377' 000000  000040
  1481  001400' 377700  740000          MWORD   <ADDR=3777,CJS,J=74>            ; call subroutine       (3777)
  1482  001401' 000000  000020
  1483  001402' 400047  770000          MWORD   <ADDR=4000,JMAP,J=4777>         ; go to next address    (4000)
  1484  001403' 000000  000040
  1485  001404' 477700  740000          MWORD   <ADDR=4777,CJS,J=74>            ; call subroutine       (4777)
  1486  001405' 000000  000020
  1487  001406' 500053  770000          MWORD   <ADDR=5000,JMAP,J=5377>         ; go to next address    (5000)
  1488  001407' 000000  000040
  1489  001410' 537700  740000          MWORD   <ADDR=5377,CJS,J=74>            ; call subroutine       (5377)
  1490  001411' 000000  000020
  1491  001412' 540055  770000          MWORD   <ADDR=5400,JMAP,J=5577>         ; go to next address    (5400)
  1492  001413' 000000  000040
  1493  001414' 557700  740000          MWORD   <ADDR=5577,CJS,J=74>            ; call subroutine       (5577)
  1494  001415' 000000  000020
  1495  001416' 560056  770000          MWORD   <ADDR=5600,JMAP,J=5677>         ; go to next address    (5600)
  1496  001417' 000000  000040
  1497  001420' 567700  740000          MWORD   <ADDR=5677,CJS,J=74>            ; call subroutine       (5677)
  1498  001421' 000000  000020
  1499  001422' 570057  370000          MWORD   <ADDR=5700,JMAP,J=5737>         ; go to next address    (5700)
  1500  001423' 000000  000040
  1501  001424' 573700  740000          MWORD   <ADDR=5737,CJS,J=74>            ; call subroutine       (5737)
  1502  001425' 000000  000020
  1503  001426' 574057  570000          MWORD   <ADDR=5740,JMAP,J=5757>         ; go to next address    (5740)
  1504  001427' 000000  000040
  1505  001430' 575700  740000          MWORD   <ADDR=5757,CJS,J=74>            ; call subroutine       (5757)
  1506  001431' 000000  000020
  1507  001432' 576057  670000          MWORD   <ADDR=5760,JMAP,J=5767>         ; go to next address    (5760)
  1508  001433' 000000  000040
  1509  001434' 576700  740000          MWORD   <ADDR=5767,CJS,J=74>            ; call subroutine       (5767)
  1510  001435' 000000  000020
  1511  001436' 577057  730000          MWORD   <ADDR=5770,JMAP,J=5773>         ; go to next address    (5770)
  1512  001437' 000000  000040
  1513  001440' 577300  740000          MWORD   <ADDR=5773,CJS,J=74>            ; call subroutine       (5773)
  1514  001441' 000000  000020
  1515  001442' 577400  740000          MWORD   <ADDR=5774,CJS,J=74>            ; call subroutine       (5774)
  1516  001443' 000000  000020
  1517  001444' 577500  740000          MWORD   <ADDR=5775,CJS,J=74>            ; call subroutine       (5775)
  1518  001445' 000000  000020
  1519  001446' 577600  740000          MWORD   <ADDR=5776,CJS,J=74>            ; call subroutine       (5776)
  1520  001447' 000000  000020
  1521  001450' 577724  000000          MWORD   <ADDR=5777,JMAP,J=2400>         ; done                  (5777)
  1522  001451' 000000  000040
  1523  001452' 240024  000000          MWORD   <ADDR=2400,JMAP,J=2400>         ; error                 (2400)
  1524  001453' 000000  000040
```

```
1525
1526  001454' 007300  000000              MWORD   <ADDR=73,JZ>                  ; initialize stack      (73)
1527  001455' 000000  000000
1528  001456' 007400  000000              MWORD   <ADDR=74,CRTN>                ; return                (74)
1529  001457' 000000  000240
1530  001460' 777777  777777              -1
```

```
                                      ;#*********************************************************************
                                      ;* Test 16 - Stack location #2
                                      ;
                                      ; Description:  Verify that the second stack  location  can  hold
                                      ;               various data patterns.
                                      ;
                                      ; Procedure:    Clear Port
                                      ;               Load test microcode and set RAR to 73
                                      ;               Set MPRUN/SINGLE STEP (execute JZ)
                                      ;               Set MPRUN/SINGLE STEP 63 times
                                      ;               Read LAR, and verify address = 5777
                                      ;
                                      ; Failure:      ---
                                      ;#*********************************************************************

                                      ; Test data

001461' 254 00 0 00 001472'           TSTS16: JRST    TG16                    ; go start test
001462' 240401  000016                        SEQ!MPROC!NDMP!ZSEQ!16          ; test mask
001463' 001520' 010660'                       T16M,,[ASCIZ ^Stack location #2^]
001464' 010442' 010517'                       [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
001465' 000000  001653'                       TSTS17                          ; failure test table
001466' 000000  002047'                       TSTS20                          ;   ...
001467' 000000  002237'                       TSTS21
001470' 000000  002433'                       TSTS22
001471' 777777  777777                        -1

                                      ; Start test

001472' 201 00 0 00 000000'           TG16:   MOVEI   Z3                      ; get address of module start
001473' 260 17 0 00 001301*                   GO      TRACE                   ; handle trace output
001474' 201 01 0 00 001520'                   MOVEI   1,T16M                  ; set up microcode address
001475' 260 17 0 00 001303*                   GO      TLOAD                   ; load/verify it
001476' 263 17 0 00 000000                    RTN                             ; failed - exit test

                                      ; Initialization

001477' 260 17 0 00 001305*           TL16:   GO      IPACLR                  ; clear port
001500' 402 00 0 00 001306*                   SETZM   TSTSUB                  ; initialize subtest number
001501' 201 06 0 00 001514'                   MOVEI   6,TS16                  ; get sstep table address

                                      ; Loop on single step table entries

001502' 400 15 0 00 000000            TA16:   SETZ    ERFLG,                  ; clear error flag
001503' 260 17 0 00 001311*                   GO      SEXEC                   ; execute table entry
001504' 254 00 0 00 001513'                   JRST    TX16                    ; end of sstep table
001505' 254 00 0 00 001502'                   JRST    TA16                    ; keep looping after call
001506' 474 15 0 00 000000                    SETO    ERFLG,                  ; error - end address incorrect
```

```

                                ; Handle error printouts and scope looping

001507' 027 00 0 00 001516'             SCOPER  MA16                    ; print error message
001510' 254 00 0 00 001477'             JRST    TL16                    ; loop on error
001511' 254 00 0 00 001513'             JRST    TX16                    ; altmode exit
001512' 322 15 0 00 001502'             JUMPE   ERFLG,TA16              ; do next sstep table entry

                                ; End of test

001513' 263 17 0 00 000000      TX16:   RTN                             ; return

                                ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

001514' 107700  735777          TS16:   STABLE  (SSSTRT,^D63,73,5777)
001515' 000000  000000                  STABLE  (SSLAST)

                                ; Error messages

001516' 140000  010664'         MA16:   MSG!TXNOT![ASCIZ /Stack location #2 broken/]
001517' 000000000000#                   LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                ; Test microcode

001520' 000000  020000          T16M:   MWORD   <ADDR=0,CJS,J=2>                ; call subroutine       (0)
001521' 000000  000020
001522' 000100  010000                  MWORD   <JMAP,J=1>                      ; error loop            (1)
001523' 000000  000040
001524' 000200  740000                  MWORD   <CJS,J=74>                      ; call subroutine       (2)
001525' 000000  000020
001526' 000300  740000                  MWORD   <CJS,J=74>                      ; call subroutine       (3)
001527' 000000  000020
001530' 000400  070000                  MWORD   <JMAP,J=7>                      ; go to next address    (4)
001531' 000000  000040
001532' 000700  740000                  MWORD   <ADDR=7,CJS,J=74>               ; call subroutine       (7)
001533' 000000  000020
001534' 001000  170000                  MWORD   <ADDR=10,JMAP,J=17>             ; go to next address    (10)
001535' 000000  000040
001536' 001700  740000                  MWORD   <ADDR=17,CJS,J=74>              ; call subroutine       (17)
001537' 000000  000020
001540' 002000  370000                  MWORD   <ADDR=20,JMAP,J=37>             ; go to next address    (20)
001541' 000000  000040
001542' 003700  740000                  MWORD   <ADDR=37,CJS,J=74>              ; call subroutine       (37)
001543' 000000  000020
001544' 004000  770000                  MWORD   <ADDR=40,JMAP,J=77>             ; go to next address    (40)
001545' 000000  000040
001546' 007700  740000                  MWORD   <ADDR=77,CJS,J=74>              ; call subroutine       (77)
001547' 000000  000020
001550' 010001  770000                  MWORD   <ADDR=100,JMAP,J=177>           ; go to next address    (100)
001551' 000000  000040
001552' 017700  740000                  MWORD   <ADDR=177,CJS,J=74>             ; call subroutine       (177)
001553' 000000  000020
001554' 020003  770000                  MWORD   <ADDR=200,JMAP,J=377>           ; go to next address    (200)
001555' 000000  000040
001556' 037700  740000                  MWORD   <ADDR=377,CJS,J=74>             ; call subroutine       (377)
```

```
001557' 000000  000020
001560' 040007  770000          MWORD   <ADDR=400,JMAP,J=777>           ; go to next address     (400)
001561' 000000  000040
001562' 077700  740000          MWORD   <ADDR=777,CJS,J=74>             ; call subroutine        (777)
001563' 000000  000020
001564' 100017  770000          MWORD   <ADDR=1000,JMAP,J=1777>         ; go to next address     (1000)
001565' 000000  000040
001566' 177700  740000          MWORD   <ADDR=1777,CJS,J=74>            ; call subroutine        (1777)
001567' 000000  000020
001570' 200037  770000          MWORD   <ADDR=2000,JMAP,J=3777>         ; go to next address     (2000)
001571' 000000  000040
001572' 377700  740000          MWORD   <ADDR=3777,CJS,J=74>            ; call subroutine        (3777)
001573' 000000  000020
001574' 400047  770000          MWORD   <ADDR=4000,JMAP,J=4777>         ; go to next address     (4000)
001575' 000000  000040
001576' 477700  740000          MWORD   <ADDR=4777,CJS,J=74>            ; call subroutine        (4777)
001577' 000000  000020
001600' 500053  770000          MWORD   <ADDR=5000,JMAP,J=5377>         ; go to next address     (5000)
001601' 000000  000040
001602' 537700  740000          MWORD   <ADDR=5377,CJS,J=74>            ; call subroutine        (5377)
001603' 000000  000020
001604' 540055  770000          MWORD   <ADDR=5400,JMAP,J=5577>         ; go to next address     (5400)
001605' 000000  000040
001606' 557700  740000          MWORD   <ADDR=5577,CJS,J=74>            ; call subroutine        (5577)
001607' 000000  000020
001610' 560056  [illegible]0000 MWORD   <ADDR=5600,JMAP,J=5677>         ; go to next address     (5600)
001611' 000000  000040
001612' 567700  740000          MWORD   <ADDR=5677,CJS,J=74>            ; call subroutine        (5677)
001613' 000000  000020
001614' 570057  370000          MWORD   <ADDR=5700,JMAP,J=5737>         ; go to next address     (5700)
001615' 000000  000040
001616' 573700  740000          MWORD   <ADDR=5737,CJS,J=74>            ; call subroutine        (5737)
001617' 000000  000020
001620' 574057  570000          MWORD   <ADDR=5740,JMAP,J=5757>         ; go to next address     (5740)
001621' 000000  000040
001622' 575700  740000          MWORD   <ADDR=5757,CJS,J=74>            ; call subroutine        (5757)
001623' 000000  000020
001624' 576057  670000          MWORD   <ADDR=5760,JMAP,J=5767>         ; go to next address     (5760)
001625' 000000  000040
001626' 576700  740000          MWORD   <ADDR=5767,CJS,J=74>            ; call subroutine        (5767)
001627' 000000  000020
001630' 577057  730000          MWORD   <ADDR=5770,JMAP,J=5773>         ; go to next address     (5770)
001631' 000000  000040
001632' 577300  740000          MWORD   <ADDR=5773,CJS,J=74>            ; call subroutine        (5773)
001633' 000000  000020
001634' 577400  740000          MWORD   <ADDR=5774,CJS,J=74>            ; call subroutine        (5774)
001635' 000000  000020
001636' 577500  740000          MWORD   <ADDR=5775,CJS,J=74>            ; call subroutine        (5775)
001637' 000000  000020
001640' 577600  740000          MWORD   <ADDR=5776,CJS,J=74>            ; call subroutine        (5776)
001641' 000000  000020
001642' 577724  000000          MWORD   <ADDR=5777,JMAP,J=2400>         ; done                   (5777)
001643' 000000  000040
001644' 240024  000000          MWORD   <ADDR=2400,JMAP,J=2400>         ; error                  (2400)
001645' 000000  000040
```

```

001646' 007300  000000          MWORD   <ADDR=73,JZ>                    ; initialize stack      (73)
001647' 000000  000000
001650' 007400  000000          MWORD   <ADDR=74,CRTN>                  ; return                (74)
001651' 000000  000240
001652' 777777  777777          -1
```

```
                                        ;#****************************************************************
                                        ;* Test 17 - Stack location #3
                                        ;
                                        ; Description:  Verify that the third  stack  location  can  hold
                                        ;               various data patterns.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode and set RAR to 73
                                        ;               Set MPRUN/SINGLE STEP (execute JZ)
                                        ;               Set MPRUN/SINGLE STEP 61 times
                                        ;               Read LAR, and verify address = 5777
                                        ;
                                        ; Failure:      ---
                                        ;#****************************************************************

                                        ; Test data

001653' 254 00 0 00 001664'             TSTS17: JRST    TG17                    ; go start test
001654' 240401  000017                          SEQ!MPROC!NDMP!ZSEQ!17          ; test mask
001655' 001712' 010671'                         T17M,,[ASCIZ ^Stack location #3^]
001656' 010442' 010517'                         [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
001657' 000000  002047'                         TSTS20                          ; failure test table
001660' 000000  002237'                         TSTS21                          ;   ...
001661' 000000  002433'                         TSTS22
001662' 000000  002511'                         TSTS23
001663' 777777  777777                          -1

                                        ; Start test

001664' 201 00 0 00 000000'             TG17:   MOVEI   Z3                      ; get address of module start
001665' 260 17 0 00 001473*                     GO      TRACE                   ; handle trace output
001666' 201 01 0 00 001712'                     MOVEI   1,T17M                  ; set up microcode address
001667' 260 17 0 00 001475*                     GO      TLOAD                   ; load/verify it
001670' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

001671' 260 17 0 00 001477*             TL17:   GO      IPACLR                  ; clear port
001672' 402 00 0 00 001500*                     SETZM   TSTSUB                  ; initialize subtest number
001673' 201 06 0 00 001706'                     MOVEI   6,TS17                  ; get sstep table address

                                        ; Loop on single step table entries

001674' 400 15 0 00 000000              TA17:   SETZ    ERFLG,                  ; clear error flag
001675' 260 17 0 00 001503*                     GO      SEXEC                   ; execute table entry
001676' 254 00 0 00 001705'                     JRST    TX17                    ; end of sstep table
001677' 254 00 0 00 001674'                     JRST    TA17                    ; keep looping after call
001700' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

001701' 027 00 0 00 001710'                     SCOPER  MA17                    ; print error message
001702' 254 00 0 00 001671'                     JRST    TL17                    ; loop on error
001703' 254 00 0 00 001705'                     JRST    TX17                    ; altmode exit
001704' 322 15 0 00 001674'                     JUMPE   ERFLG,TA17              ; do next sstep table entry

                                        ; End of test

001705' 263 17 0 00 000000              TX17:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

001706' 107600  735777                  TS17:   STABLE  (SSSTRT,^D62,73,5777)
001707' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

001710' 140000  010675'                 MA17:   MSG!TXNOT![ASCIZ /Stack location #3 broken/]
001711' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

001712' 000000  020000                  T17M:   MWORD   <ADDR=0,CJS,J=2>                ; call subroutine       (0)
001713' 000000  000020
001714' 000100  010000                          MWORD   <JMAP,J=1>                      ; error loop            (1)
001715' 000000  000040
001716' 000200  040000                          MWORD   <CJS,J=4>                       ; call subroutine       (2)
001717' 000000  000020
001720' 000300  030000                          MWORD   <JMAP,J=3>                      ; error loop            (3)
001721' 000000  000040
001722' 000400  740000                          MWORD   <CJS,J=74>                      ; call subroutine       (4)
001723' 000000  000020
001724' 000500  070000                          MWORD   <JMAP,J=7>                      ; go to next address    (5)
001725' 000000  000040
001726' 000700  740000                          MWORD   <ADDR=7,CJS,J=74>               ; call subroutine       (7)
001727' 000000  000020
001730' 001000  170000                          MWORD   <ADDR=10,JMAP,J=17>             ; go to next address    (10)
001731' 000000  000040
001732' 001700  740000                          MWORD   <ADDR=17,CJS,J=74>              ; call subroutine       (17)
001733' 000000  000020
001734' 002000  370000                          MWORD   <ADDR=20,JMAP,J=37>             ; go to next address    (20)
001735' 000000  000040
001736' 003700  740000                          MWORD   <ADDR=37,CJS,J=74>              ; call subroutine       (37)
001737' 000000  000020
001740' 004000  770000                          MWORD   <ADDR=40,JMAP,J=77>             ; go to next address    (40)
001741' 000000  000040
001742' 007700  740000                          MWORD   <ADDR=77,CJS,J=74>              ; call subroutine       (77)
001743' 000000  000020
001744' 010001  770000                          MWORD   <ADDR=100,JMAP,J=177>           ; go to next address    (100)
001745' 000000  000040
001746' 017700  740000                          MWORD   <ADDR=177,CJS,J=74>             ; call subroutine       (177)
001747' 000000  000020
001750' 020003  770000                          MWORD   <ADDR=200,JMAP,J=377>           ; go to next address    (200)
```

```
1800  001751' 000000  000040
1801  001752' 037700  740000          MWORD   <ADDR=377,CJS,J=74>             ; call subroutine        (377)
1802  001753' 000000  000020
1803  001754' 040007  770000          MWORD   <ADDR=400,JMAP,J=777>           ; go to next address     (400)
1804  001755' 000000  000040
1805  001756' 077700  740000          MWORD   <ADDR=777,CJS,J=74>             ; call subroutine        (777)
1806  001757' 000000  000020
1807  001760' 100017  770000          MWORD   <ADDR=1000,JMAP,J=1777>         ; go to next address     (1000)
1808  001761' 000000  000040
1809  001762' 177700  740000          MWORD   <ADDR=1777,CJS,J=74>            ; call subroutine        (1777)
1810  001763' 000000  000020
1811  001764' 200037  770000          MWORD   <ADDR=2000,JMAP,J=3777>         ; go to next address     (2000)
1812  001765' 000000  000040
1813  001766' 377700  740000          MWORD   <ADDR=3777,CJS,J=74>            ; call subroutine        (3777)
1814  001767' 000000  000020
1815  001770' 400047  770000          MWORD   <ADDR=4000,JMAP,J=4777>         ; go to next address     (4000)
1816  001771' 000000  000040
1817  001772' 477700  740000          MWORD   <ADDR=4777,CJS,J=74>            ; call subroutine        (4777)
1818  001773' 000000  000020
1819  001774' 500053  770000          MWORD   <ADDR=5000,JMAP,J=5377>         ; go to next address     (5000)
1820  001775' 000000  000040
1821  001776' 537700  740000          MWORD   <ADDR=5377,CJS,J=74>            ; call subroutine        (5377)
1822  001777' 000000  000020
1823  002000' 540055  770000          MWORD   <ADDR=5400,JMAP,J=5577>         ; go to next address     (5400)
1824  002001' 000000  000040
1825  002002' 557700  740000          MWORD   <ADDR=5577,CJS,J=74>            ; call subroutine        (5577)
1826  002003' 000000  000020
1827  002004' 560056  770000          MWORD   <ADDR=5600,JMAP,J=5677>         ; go to next address     (5600)
1828  002005' 000000  000040
1829  002006' 567700  740000          MWORD   <ADDR=5677,CJS,J=74>            ; call subroutine        (5677)
1830  002007' 000000  000020
1831  002010' 570057  370000          MWORD   <ADDR=5700,JMAP,J=5737>         ; go to next address     (5700)
1832  002011' 000000  000040
1833  002012' 573700  740000          MWORD   <ADDR=5737,CJS,J=74>            ; call subroutine        (5737)
1834  002013' 000000  000020
1835  002014' 574057  570000          MWORD   <ADDR=5740,JMAP,J=5757>         ; go to next address     (5740)
1836  002015' 000000  000040
1837  002016' 575700  740000          MWORD   <ADDR=5757,CJS,J=74>            ; call subroutine        (5757)
1838  002017' 000000  000020
1839  002020' 576057  670000          MWORD   <ADDR=5760,JMAP,J=5767>         ; go to next address     (5760)
1840  002021' 000000  000040
1841  002022' 576700  740000          MWORD   <ADDR=5767,CJS,J=74>            ; call subroutine        (5767)
1842  002023' 000000  000020
1843  002024' 577057  730000          MWORD   <ADDR=5770,JMAP,J=5773>         ; go to next address     (5770)
1844  002025' 000000  000040
1845  002026' 577300  740000          MWORD   <ADDR=5773,CJS,J=74>            ; call subroutine        (5773)
1846  002027' 000000  000020
1847  002030' 577400  740000          MWORD   <ADDR=5774,CJS,J=74>            ; call subroutine        (5774)
1848  002031' 000000  000020
1849  002032' 577500  740000          MWORD   <ADDR=5775,CJS,J=74>            ; call subroutine        (5775)
1850  002033' 000000  000020
1851  002034' 577600  740000          MWORD   <ADDR=5776,CJS,J=74>            ; call subroutine        (5776)
1852  002035' 000000  000020
1853  002036' 577724  000000          MWORD   <ADDR=5777,JMAP,J=2400>         ; done                   (5777)
1854  002037' 000000  000040
```

```
 1855  002040' 240024  000000               MWORD   <ADDR=2400,JMAP,J=2400>         ; error              (2400)
 1856  002041' 000000  000040
 1857
 1858  002042' 007300  000000               MWORD   <ADDR=73,JZ>                    ; initialize stack   (73)
 1859  002043' 000000  000000
 1860  002044' 007400  000000               MWORD   <ADDR=74,CRTN>                  ; return             (74)
 1861  002045' 000000  000240
 1862  002046' 777777  777777               -1
```

```

                                        ;#*****************************************************************
                                        ;* Test 20 - Stack location #4
                                        ;
                                        ; Description:  Verify that the fourth stack  location  can  hold
                                        ;               various data patterns.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode and set RAR to 73
                                        ;               Set MPRUN/SINGLE STEP (execute JZ)
                                        ;               Set MPRUN/SINGLE STEP 61 times
                                        ;               Read LAR, and verify address = 5777
                                        ;
                                        ; Failure:      ---
                                        ;#*****************************************************************

                                        ; Test data

002047' 254 00 0 00 002060'             TSTS20: JRST    TG20                    ; go start test
002050' 240401  000020                          SEQ!MPROC!NDMP!ZSEQ!20          ; test mask
002051' 002106' 010702'                         T20M,,[ASCIZ ^Stack location #4^]
002052' 010442' 010517'                         [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
002053' 000000  002237'                         TSTS21                          ; failure test table
002054' 000000  002433'                         TSTS22                          ;   ...
002055' 000000  002511'                         TSTS23
002056' 000000  002571'                         TSTS24
002057' 777777  777777                          -1

                                        ; Start test

002060' 201 00 0 00 000000'             TG20:   MOVEI   Z3                      ; get address of module start
002061' 260 17 0 00 001665*                     GO      TRACE                   ; handle trace output
002062' 201 01 0 00 002106'                     MOVEI   1,T20M                  ; set up microcode address
002063' 260 17 0 00 001667*                     GO      TLOAD                   ; load/verify it
002064' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

002065' 260 17 0 00 001671*             TL20:   GO      IPACLR                  ; clear port
002066' 402 00 0 00 001672*                     SETZM   TSTSUB                  ; initialize subtest number
002067' 201 06 0 00 002102'                     MOVEI   6,TS20                  ; get sstep table address

                                        ; Loop on single step table entries

002070' 400 15 0 00 000000              TA20:   SETZ    ERFLG,                  ; clear error flag
002071' 260 17 0 00 001675*                     GO      SEXEC                   ; execute table entry
002072' 254 00 0 00 002101'                     JRST    TX20                    ; end of sstep table
002073' 254 00 0 00 002070'                     JRST    TA20                    ; keep looping after call
002074' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

002075' 027 00 0 00 002104'                     SCOPER  MA20                    ; print error message
002076' 254 00 0 00 002065'                     JRST    TL20                    ; loop on error
002077' 254 00 0 00 002101'                     JRST    TX20                    ; altmode exit
002100' 322 15 0 00 002070'                     JUMPE   ERFLG,TA20              ; do next sstep table entry

                                        ; End of test

002101' 263 17 0 00 000000              TX20:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

002102' 107400  735777                  TS20:   STABLE  (SSSTRT,^D60,73,5777)
002103' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

002104' 140000  010706'                 MA20:   MSG!TXNOT![ASCIZ /Stack location #4 broken/]
002105' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

002106' 000000  020000                  T20M:   MWORD   <ADDR=0,CJS,J=2>                ; call subroutine       (0)
002107' 000000  000020
002110' 000100  010000                          MWORD   <JMAP,J=1>                      ; error loop            (1)
002111' 000000  000040
002112' 000200  040000                          MWORD   <CJS,J=4>                       ; call subroutine       (2)
002113' 000000  000020
002114' 000300  030000                          MWORD   <JMAP,J=3>                      ; error loop            (3)
002115' 000000  000040
002116' 000400  070000                          MWORD   <CJS,J=7>                       ; call subroutine       (4)
002117' 000000  000020
002120' 000500  050000                          MWORD   <JMAP,J=5>                      ; go to next address    (5)
002121' 000000  000040
002122' 000700  740000                          MWORD   <ADDR=7,CJS,J=74>               ; call subroutine       (7)
002123' 000000  000020
002124' 001000  170000                          MWORD   <ADDR=10,JMAP,J=17>             ; go to next address    (10)
002125' 000000  000040
002126' 001700  740000                          MWORD   <ADDR=17,CJS,J=74>              ; call subroutine       (17)
002127' 000000  000020
002130' 002000  370000                          MWORD   <ADDR=20,JMAP,J=37>             ; go to next address    (20)
002131' 000000  000040
002132' 003700  740000                          MWORD   <ADDR=37,CJS,J=74>              ; call subroutine       (37)
002133' 000000  000020
002134' 004000  770000                          MWORD   <ADDR=40,JMAP,J=77>             ; go to next address    (40)
002135' 000000  000040
002136' 007700  740000                          MWORD   <ADDR=77,CJS,J=74>              ; call subroutine       (77)
002137' 000000  000020
002140' 010001  770000                          MWORD   <ADDR=100,JMAP,J=177>           ; go to next address    (100)
002141' 000000  000040
002142' 017700  740000                          MWORD   <ADDR=177,CJS,J=74>             ; call subroutine       (177)
002143' 000000  000020
002144' 020003  770000                          MWORD   <ADDR=200,JMAP,J=377>           ; go to next address    (200)
```

```
002145' 000000 000040
002146' 037700 740000          MWORD   <ADDR=377,CJS,J=74>            ; call subroutine       (377)
002147' 000000 000020
002150' 040007 770000          MWORD   <ADDR=400,JMAP,J=777>          ; go to next address    (400)
002151' 000000 000040
002152' 077700 740000          MWORD   <ADDR=777,CJS,J=74>            ; call subroutine       (777)
002153' 000000 000020
002154' 100017 770000          MWORD   <ADDR=1000,JMAP,J=1777>        ; go to next address    (1000)
002155' 000000 000040
002156' 177700 740000          MWORD   <ADDR=1777,CJS,J=74>           ; call subroutine       (1777)
002157' 000000 000020
002160' 200037 770000          MWORD   <ADDR=2000,JMAP,J=3777>        ; go to next address    (2000)
002161' 000000 000040
002162' 377700 740000          MWORD   <ADDR=3777,CJS,J=74>           ; call subroutine       (3777)
002163' 000000 000020
002164' 400047 770000          MWORD   <ADDR=4000,JMAP,J=4777>        ; go to next address    (4000)
002165' 000000 000040
002166' 477700 740000          MWORD   <ADDR=4777,CJS,J=74>           ; call subroutine       (4777)
002167' 000000 000020
002170' 500053 770000          MWORD   <ADDR=5000,JMAP,J=5377>        ; go to next address    (5000)
002171' 000000 000040
0021[illegible]' 537700 740000          MWORD   <ADDR=5377,CJS,J=74>           ; call subroutine       (5377)
0021[illegible]3' 000000 000020
002174' 540055 770000          MWORD   <ADDR=5400,JMAP,J=5577>        ; go to next address    (5400)
002175' 000000 000040
002176' 557700 740000          MWORD   <ADDR=5577,CJS,J=74>           ; call subroutine       (5577)
002177' 000000 000020
002200' 560056 770000          MWORD   <ADDR=5600,JMAP,J=5677>        ; go to next address    (5600)
002201' 000000 000040
002202' 567700 740000          MWORD   <ADDR=5677,CJS,J=74>           ; call subroutine       (5677)
002203' 000000 000020
002204' 570057 370000          MWORD   <ADDR=5700,JMAP,J=5737>        ; go to next address    (5700)
002205' 000000 000040
002206' 573700 740000          MWORD   <ADDR=5737,CJS,J=74>           ; call subroutine       (5737)
002207' 000000 000020
002210' 574057 570000          MWORD   <ADDR=5740,JMAP,J=5757>        ; go to next address    (5740)
002211' 000000 000040
002212' 575700 740000          MWORD   <ADDR=5757,CJS,J=74>           ; call subroutine       (5757)
002213' 000000 000020
002214' 576057 670000          MWORD   <ADDR=5760,JMAP,J=5767>        ; go to next address    (5760)
002215' 000000 000040
002216' 576700 740000          MWORD   <ADDR=5767,CJS,J=74>           ; call subroutine       (5767)
002217' 000000 000020
002220' 577057 730000          MWORD   <ADDR=5770,JMAP,J=5773>        ; go to next address    (5770)
002221' 000000 000040
002222' 577300 740000          MWORD   <ADDR=5773,CJS,J=74>           ; call subroutine       (5773)
002223' 000000 000020
002224' 577400 740000          MWORD   <ADDR=5774,CJS,J=74>           ; call subroutine       (5774)
002225' 000000 000020
002226' 577500 740000          MWORD   <ADDR=5775,CJS,J=74>           ; call subroutine       (5775)
002227' 000000 000020
002230' 577600 740000          MWORD   <ADDR=5776,CJS,J=74>           ; call subroutine       (5776)
002231' 000000 000020

002232' 007300 000000          MWORD   <ADDR=73,JZ>                   ; initialize stack      (73)
```

```
  2022  002233' 000000  000000
  2023  002234' 007400  000000              MWORD   <ADDR=74,CRTN>                  ; return                (74)
  2024  002235' 000000  000240
  2025  002236' 777777  777777              -1
```

```

                                        ;#*********************************************************************
                                        ;* Test 21 - Stack location #5
                                        ;
                                        ; Description:  Verify that the fifth  stack  location  can  hold
                                        ;               various data patterns.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode and set RAR to 73
                                        ;               Set MPRUN/SINGLE STEP (execute JZ)
                                        ;               Set MPRUN/SINGLE STEP 61 times
                                        ;               Read LAR, and verify address = 5777
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

002237' 254 00 0 00 002250'             TSTS21: JRST    TG21                    ; go start test
002240' 240401  000021                          SEQ!MPROC!NDMP!ZSEQ!21          ; test mask
002241' 002276' 010713'                         T21M,,[ASCIZ ^Stack location #5^]
002242' 010442' 010517'                         [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
002243' 000000  002433'                         TSTS22                          ; failure test table
002244' 000000  002511'                         TSTS23                          ;   ...
002245' 000000  002571'                         TSTS24
002246' 000000  002653'                         TSTS25
002247' 777777  777777                          -1

                                        ; Start test

002250' 201 00 0 00 000000'             TG21:   MOVEI   Z3                      ; get address of module start
002251' 260 17 0 00 002061*                     GO      TRACE                   ; handle trace output
002252' 201 01 0 00 002276'                     MOVEI   1,T21M                  ; set up microcode address
002253' 260 17 0 00 002063*                     GO      TLOAD                   ; load/verify it
002254' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

002255' 260 17 0 00 002065*             TL21:   GO      IPACLR                  ; clear port
002256' 402 00 0 00 002066*                     SETZM   TSTSUB                  ; initialize subtest number
002257' 201 06 0 00 002272'                     MOVEI   6,TS21                  ; get sstep table address

                                        ; Loop on single step table entries

002260' 400 15 0 00 000000              TA21:   SETZ    ERFLG,                  ; clear error flag
002261' 260 17 0 00 002071*                     GO      SEXEC                   ; execute table entry
002262' 254 00 0 00 002271'                     JRST    TX21                    ; end of sstep table
002263' 254 00 0 00 002260'                     JRST    TA21                    ; keep looping after call
002264' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

        002265' 027 00 0 00 002274'             SCOPER  MA21                    ; print error message
        002266' 254 00 0 00 002255'             JRST    TL21                    ; loop on error
        002267' 254 00 0 00 002271'             JRST    TX21                    ; altmode exit
        002270' 322 15 0 00 002260'             JUMPE   ERFLG,TA21              ; do next sstep table entry

                                        ; End of test

        002271' 263 17 0 00 000000      TX21:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

        002272' 107200  735777          TS21:   STABLE  (SSSTRT,^D58,73,5777)
        002273' 000000  000000                  STABLE  (SSLAST)

                                        ; Error messages

        002274' 140000  010717'         MA21:   MSG!TXNOT![ASCIZ /Stack location #5 broken/]
        002275' 000000000000#                   LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

        002276' 000000  020000          T21M:   MWORD   <ADDR=0,CJS,J=2>                ; call subroutine       (0)
        002277' 000000  000020
        002300' 000100  010000                  MWORD   <JMAP,J=1>                      ; error loop            (1)
        002301' 000000  000040
        002302' 000200  040000                  MWORD   <CJS,J=4>                       ; call subroutine       (2)
        002303' 000000  000020
        002304' 000300  030000                  MWORD   <JMAP,J=3>                      ; error loop            (3)
        002305' 000000  000040
        002306' 000400  070000                  MWORD   <CJS,J=7>                       ; call subroutine       (4)
        002307' 000000  000020
        002310' 000500  050000                  MWORD   <JMAP,J=5>                      ; go to next address    (5)
        002311' 000000  000040
        002312' 000700  170000                  MWORD   <ADDR=7,CJS,J=17>               ; call subroutine       (7)
        002313' 000000  000020
        002314' 001000  100000                  MWORD   <ADDR=10,JMAP,J=10>             ; go to next address    (10)
        002315' 000000  000040
        002316' 001700  740000                  MWORD   <ADDR=17,CJS,J=74>              ; call subroutine       (17)
        002317' 000000  000020
        002320' 002000  370000                  MWORD   <ADDR=20,JMAP,J=37>             ; go to next address    (20)
        002321' 000000  000040
        002322' 003700  740000                  MWORD   <ADDR=37,CJS,J=74>              ; call subroutine       (37)
        002323' 000000  000020
        002324' 004000  770000                  MWORD   <ADDR=40,JMAP,J=77>             ; go to next address    (40)
        002325' 000000  000040
        002326' 007700  740000                  MWORD   <ADDR=77,CJS,J=74>              ; call subroutine       (77)
        002327' 000000  000020
        002330' 010001  770000                  MWORD   <ADDR=100,JMAP,J=177>           ; go to next address    (100)
        002331' 000000  000040
        002332' 017700  740000                  MWORD   <ADDR=177,CJS,J=74>             ; call subroutine       (177)
        002333' 000000  000020
        002334' 020003  770000                  MWORD   <ADDR=200,JMAP,J=377>           ; go to next address    (200)
```

```
 2130  002335' 000000  000040
 2131  002336' 037700  740000          MWORD   <ADDR=377,CJS,J=74>          ; call subroutine       (377)
 2132  002337' 000000  000020
 2133  002340' 040007  770000          MWORD   <ADDR=400,JMAP,J=777>        ; go to next address    (400)
 2134  002341' 000000  000040
 2135  002342' 077700  740000          MWORD   <ADDR=777,CJS,J=74>          ; call subroutine       (777)
 2136  002343' 000000  000020
 2137  002344' 100017  770000          MWORD   <ADDR=1000,JMAP,J=1777>      ; go to next address    (1000)
 2138  002345' 000000  000040
 2139  002346' 177700  740000          MWORD   <ADDR=1777,CJS,J=74>         ; call subroutine       (1777)
 2140  002347' 000000  000020
 2141  002350' 200037  770000          MWORD   <ADDR=2000,JMAP,J=3777>      ; go to next address    (2000)
 2142  002351' 000000  000040
 2143  002352' 377700  740000          MWORD   <ADDR=3777,CJS,J=74>         ; call subroutine       (3777)
 2144  002353' 000000  000020
 2145  002354' 400047  770000          MWORD   <ADDR=4000,JMAP,J=4777>      ; go to next address    (4000)
 2146  002355' 000000  000040
 2147  002356' 477700  740000          MWORD   <ADDR=4777,CJS,J=74>         ; call subroutine       (4777)
 2148  002357' 000000  000020
 2149  002360' 500053  770000          MWORD   <ADDR=5000,JMAP,J=5377>      ; go to next address    (5000)
 2150  002361' 000000  000040
 2151  002362' 537700  740000          MWORD   <ADDR=5377,CJS,J=74>         ; call subroutine       (5377)
 2152  002363' 000000  000020
 2153  002364' 540055  770000          MWORD   <ADDR=5400,JMAP,J=5577>      ; go to next address    (5400)
 2154  002365' 000000  000040
 2155  002366' 557700  740000          MWORD   <ADDR=5577,CJS,J=74>         ; call subroutine       (5577)
 2156  002367' 000000  000020
 2157  002370' 560056  770000          MWORD   <ADDR=5600,JMAP,J=5677>      ; go to next address    (5600)
 2158  002371' 000000  000040
 2159  002372' 567700  740000          MWORD   <ADDR=5677,CJS,J=74>         ; call subroutine       (5677)
 2160  002373' 000000  000020
 2161  002374' 570057  370000          MWORD   <ADDR=5700,JMAP,J=5737>      ; go to next address    (5700)
 2162  002375' 000000  000040
 2163  002376' 573700  740000          MWORD   <ADDR=5737,CJS,J=74>         ; call subroutine       (5737)
 2164  002377' 000000  000020
 2165  002400' 574057  570000          MWORD   <ADDR=5740,JMAP,J=5757>      ; go to next address    (5740)
 2166  002401' 000000  000040
 2167  002402' 575700  740000          MWORD   <ADDR=5757,CJS,J=74>         ; call subroutine       (5757)
 2168  002403' 000000  000020
 2169  002404' 576057  670000          MWORD   <ADDR=5760,JMAP,J=5767>      ; go to next address    (5760)
 2170  002405' 000000  000040
 2171  002406' 576700  740000          MWORD   <ADDR=5767,CJS,J=74>         ; call subroutine       (5767)
 2172  002407' 000000  000020
 2173  002410' 577057  730000          MWORD   <ADDR=5770,JMAP,J=5773>      ; go to next address    (5770)
 2174  002411' 000000  000040
 2175  002412' 577300  740000          MWORD   <ADDR=5773,CJS,J=74>         ; call subroutine       (5773)
 2176  002413' 000000  000020
 2177  002414' 577400  740000          MWORD   <ADDR=5774,CJS,J=74>         ; call subroutine       (5774)
 2178  002415' 000000  000020
 2179  002416' 577500  740000          MWORD   <ADDR=5775,CJS,J=74>         ; call subroutine       (5775)
 2180  002417' 000000  000020
 2181  002420' 577600  740000          MWORD   <ADDR=5776,CJS,J=74>         ; call subroutine       (5776)
 2182  002421' 000000  000020
 2183  002422' 577724  000000          MWORD   <ADDR=5777,JMAP,J=2400>      ; done                  (5777)
 2184  002423' 000000  000040
```

```
002424' 240024  000000          MWORD   <ADDR=2400,JMAP,J=2400>         ; error                 (2400)
002425' 000000  000040

002426' 007300  000000          MWORD   <ADDR=73,JZ>                    ; initialize stack      (73)
002427' 000000  000000
002430' 007400  000000          MWORD   <ADDR=74,CRTN>                  ; return                (74)
002431' 000000  000240
002432' 777777  777777          -1
```

```
                                        ;#**********************************************************************
                                        ;* Test 22 - Increment/Decrement Stack Pointer
                                        ;
                                        ; Description:  Verify that the 2910 stack pointer increments and
                                        ;               decrements correctly.  Do five nested  subroutine
                                        ;               calls followed by a return.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode
                                        ;               Set up RAR to start at location 1
                                        ;               Set MPRUN/SINGLE STEP (execute JZ)
                                        ;               Set MPRUN/SINGLE STEP 5 times (execute 5 CJS's)
                                        ;               Set MPRUN/SINGLE STEP (execute CRTN)
                                        ;               Read LAR, and verify address = 16
                                        ;
                                        ; Failure:      ---
                                        ;#**********************************************************************

                                        ; Test data

002433' 254 00 0 00 002444'             TSTS22: JRST    TG22                    ; go start test
002434' 240401  000022                          SEQ!MPROC!NDMP!ZSEQ!22          ; test mask
002435' 002472' 010724'                         T22M,,[ASCIZ ^Increment/Decrement Stack Pointer^]
002436' 010442' 010517'                         [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
002437' 000000  002511'                         TSTS23                          ; failure test table
002440' 000000  002571'                         TSTS24                          ;   ...
002441' 000000  002653'                         TSTS25
002442' 000000  002737'                         TSTS26
002443' 777777  777777                          -1

                                        ; Start test

002444' 201 00 0 00 000000'             TG22:   MOVEI   Z3                      ; get address of module start
002445' 260 17 0 00 002251*                     GO      TRACE                   ; handle trace output
002446' 201 01 0 00 002472'                     MOVEI   1,T22M                  ; set up microcode address
002447' 260 17 0 00 002253*                     GO      TLOAD                   ; load/verify it
002450' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

002451' 260 17 0 00 002255*             TL22:   GO      IPACLR                  ; clear port
002452' 402 00 0 00 002256*                     SETZM   TSTSUB                  ; initialize subtest number
002453' 201 06 0 00 002466'                     MOVEI   6,TS22                  ; get sstep table address

                                        ; Loop on single step table entries

002454' 400 15 0 00 000000              TA22:   SETZ    ERFLG,                  ; clear error flag
002455' 260 17 0 00 002261*                     GO      SEXEC                   ; execute table entry
002456' 254 00 0 00 002465'                     JRST    TX22                    ; end of sstep table
002457' 254 00 0 00 002454'                     JRST    TA22                    ; keep looping after call
002460' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

002461' 027 00 0 00 002470'                     SCOPER  MA22                    ; print error message
002462' 254 00 0 00 002451'                     JRST    TL22                    ; loop on error
002463' 254 00 0 00 002465'                     JRST    TX22                    ; altmode exit
002464' 322 15 0 00 002454'                     JUMPE   ERFLG,TA22              ; do next sstep table entry

                                        ; End of test

002465' 263 17 0 00 000000              TX22:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

002466' 100700  010016                  TS22:   STABLE  (SSSTRT,7,1,16)
002467' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

002470' 140000  010733'                 MA22:   MSG!TXNOT![ASCIZ /Stack pointer not working/]
002471' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

002472' 000000  040000                  T22M:   MWORD   <ADDR=0,CJS,J=4>                ; call subroutine       (0)
002473' 000000  000020
002474' 000100  000000                          MWORD   <JZ>                            ; initialize stack      (1)
002475' 000000  000000
002476' 000400  070000                          MWORD   <ADDR=4,CJS,J=7>                ; call subroutine       (4)
002477' 000000  000020
002500' 000700  120000                          MWORD   <ADDR=7,CJS,J=12>               ; call subroutine       (7)
002501' 000000  000020
002502' 001200  150000                          MWORD   <ADDR=12,CJS,J=15>              ; call subroutine       (12)
002503' 000000  000020
002504' 001500  200000                          MWORD   <ADDR=15,CJS,J=20>              ; call subroutine       (15)
002505' 000000  000020
002506' 002000  000000                          MWORD   <ADDR=20,CRTN>                  ; return to 16          (20)
002507' 000000  000240
002510' 777777  777777                          -1
```

```
                                        ;#*********************************************************************
                                        ;* Test 23 - Increment/Decrement Stack Pointer
                                        ;
                                        ; Description:  Verify that the 2910 stack pointer increments and
                                        ;               decrements correctly.   Do five nested subroutine
                                        ;               calls followed by two returns.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode
                                        ;               Set up RAR to start at location 1
                                        ;               Set MPRUN/SINGLE STEP (execute JZ)
                                        ;               Set MPRUN/SINGLE STEP 5 times (execute 5 CJS's)
                                        ;               Set MPRUN/SINGLE STEP (execute CRTN)
                                        ;               Set MPRUN/SINGLE STEP (execute CRTN)
                                        ;               Read LAR, and verify address = 13
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

 002511' 254 00 0 00 002522'            TSTS23: JRST    TG23                    ; go start test
 002512' 240401  000023                         SEQ!MPROC!NDMP!ZSEQ!23          ; test mask
 002513' 002550' 010724'                        T23M,,[ASCIZ ^Increment/Decrement Stack Pointer^]
 002514' 010442' 010517'                        [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
 002515' 000000  002571'                        TSTS24                          ; failure test table
 002516' 000000  002653'                        TSTS25                          ;   ...
 002517' 000000  002737'                        TSTS26
 002520' 000000  003032'                        TSTS30
 002521' 777777  777777                         -1

                                        ; Start test

 002522' 201 00 0 00 000000'            TG23:   MOVEI   Z3                      ; get address of module start
 002523' 260 17 0 00 002445*                    GO      TRACE                   ; handle trace output
 002524' 201 01 0 00 002550'                    MOVEI   1,T23M                  ; set up microcode address
 002525' 260 17 0 00 002447*                    GO      TLOAD                   ; load/verify it
 002526' 263 17 0 00 000000                     RTN                             ; failed - exit test

                                        ; Initialization

 002527' 260 17 0 00 002451*            TL23:   GO      IPACLR                  ; clear port
 002530' 402 00 0 00 002452*                    SETZM   TSTSUB                  ; initialize subtest number
 002531' 201 06 0 00 002544'                    MOVEI   6,TS23                  ; get sstep table address

                                        ; Loop on single step table entries

 002532' 400 15 0 00 000000             TA23:   SETZ    ERFLG,                  ; clear error flag
 002533' 260 17 0 00 002455*                    GO      SEXEC                   ; execute table entry
 002534' 254 00 0 00 002543'                    JRST    TX23                    ; end of sstep table
 002535' 254 00 0 00 002532'                    JRST    TA23                    ; keep looping after call
 002536' 474 15 0 00 000000                     SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

        002537' 027 00 0 00 002546'             SCOPER  MA23                    ; print error message
        002540' 254 00 0 00 002527'             JRST    TL23                    ; loop on error
        002541' 254 00 0 00 002543'             JRST    TX23                    ; altmode exit
        002542' 322 15 0 00 002532'             JUMPE   ERFLG,TA23              ; do next sstep table entry

                                        ; End of test

        002543' 263 17 0 00 000000      TX23:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

        002544' 101000  010013          TS23:   STABLE  (SSSTRT,^D8,1,13)
        002545' 000000  000000                  STABLE  (SSLAST)

                                        ; Error messages

        002546' 140000  010733'         MA23:   MSG!TXNOT![ASCIZ /Stack pointer not working/]
        002547' 000000000000#                   LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

        002550' 000000  040000          T23M:   MWORD   <ADDR=0,CJS,J=4>                ; call subroutine       (0)
        002551' 000000  000020
        002552' 000100  000000                  MWORD   <JZ>                            ; initialize stack      (1)
        002553' 000000  000000
        002554' 000400  070000                  MWORD   <ADDR=4,CJS,J=7>                ; call subroutine       (4)
        002555' 000000  000020
        002556' 000700  120000                  MWORD   <ADDR=7,CJS,J=12>               ; call subroutine       (7)
        002557' 000000  000020
        002560' 001200  150000                  MWORD   <ADDR=12,CJS,J=15>              ; call subroutine       (12)
        002561' 000000  000020
        002562' 001500  200000                  MWORD   <ADDR=15,CJS,J=20>              ; call subroutine       (15)
        002563' 000000  000020
        002564' 002000  000000                  MWORD   <ADDR=20,CRTN>                  ; return to 16          (20)
        002565' 000000  000240
        002566' 001600  000000                  MWORD   <ADDR=16,CRTN>                  ; return to 13          (16)
        002567' 000000  000240
        002570' 777777  777777                  -1
```

```
                                ;#***********************************************************
                                ;* Test 24 - Increment/Decrement Stack Pointer
                                ;
                                ; Description:  Verify that the 2910 stack pointer increments and
                                ;               decrements correctly.   Do five nested subroutine
                                ;               calls followed by three returns.
                                ;
                                ; Procedure:    Clear Port
                                ;               Load test microcode
                                ;               Set up RAR to start at location 1
                                ;               Set MPRUN/SINGLE STEP (execute JZ)
                                ;               Set MPRUN/SINGLE STEP 5 times (execute 5 CJS's)
                                ;               Set MPRUN/SINGLE STEP 3 times (execute 3 CRTN's)
                                ;               Read LAR, and verify address = 10
                                ;
                                ; Failure:      ---
                                ;#***********************************************************

                                ; Test data

002571' 254 00 0 00 002602'     TSTS24: JRST    TG24                    ; go start test
002572' 240401  000024                  SEQ!MPROC!NDMP!ZSEQ!24          ; test mask
002573' 002630' 010724'                 T24M,,[ASCIZ ^Increment/Decrement Stack Pointer^]
002574' 010442' 010517'                 [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
002575' 000000  002653'                 TSTS25                          ; failure test taple
002576' 000000  002737'                 TSTS26                          ;   ...
002577' 000000  003032'                 TSTS30
002600' 000000  003122'                 TSTS31
002601' 777777  777777                  -1

                                ; Start test

002602' 201 00 0 00 000000'     TG24:   MOVEI   Z3                      ; get address of module start
002603' 260 17 0 00 002523*             GO      TRACE                   ; handle trace output
002604' 201 01 0 00 002630'             MOVEI   1,T24M                  ; set up microcode address
002605' 260 17 0 00 002525*             GO      TLOAD                   ; load/verify it
002606' 263 17 0 00 000000              RTN                             ; failed - exit test

                                ; Initialization

002607' 260 17 0 00 002527*     TL24:   GO      IPACLR                  ; clear port
002610' 402 00 0 00 002530*             SETZM   TSTSUB                  ; initialize subtest number
002611' 201 06 0 00 002624'             MOVEI   6,TS24                  ; get sstep table address

                                ; Loop on single step table entries

002612' 400 15 0 00 000000      TA24:   SETZ    ERFLG,                  ; clear error flag
002613' 260 17 0 00 002533*             GO      SEXEC                   ; execute table entry
002614' 254 00 0 00 002623'             JRST    TX24                    ; end of sstep table
002615' 254 00 0 00 002612'             JRST    TA24                    ; keep looping after call
002616' 474 15 0 00 000000              SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

002617' 027 00 0 00 002626'                     SCOPER  MA24                    ; print error message
002620' 254 00 0 00 002607'                     JRST    TL24                    ; loop on error
002621' 254 00 0 00 002623'                     JRST    TX24                    ; altmode exit
002622' 322 15 0 00 002612'                     JUMPE   ERFLG,TA24              ; do next sstep table entry

                                        ; End of test

002623' 263 17 0 00 000000              TX24:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

002624' 101100  010010                  TS24:   STABLE  (SSSTRT,^D9,1,10)
002625' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

002626' 140000  010733'                 MA24:   MSG!TXNOT![ASCIZ /Stack pointer not working/]
002627' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

002630' 000000  040000                  T24M:   MWORD   <ADDR=0,CJS,J=4>                ; call subroutine       (0)
002631' 000000  000020
002632' 000100  000000                          MWORD   <JZ>                            ; initialize stack      (1)
002633' 000000  000000
002634' 000400  070000                          MWORD   <ADDR=4,CJS.J=7>                ; call subroutine       (4)
002635' 000000  000020
002636' 000700  120000                          MWORD   <ADDR=7,CJS,J=12>               ; call subroutine       (7)
002637' 000000  000020
002640' 001200  150000                          MWORD   <ADDR=12,CJS,J=15>              ; call subroutine       (12)
002641' 000000  000020
002642' 001500  200000                          MWORD   <ADDR=15,CJS,J=20>              ; call subroutine       (15)
002643' 000000  000020
002644' 002000  000000                          MWORD   <ADDR=20,CRTN>                  ; return to 16          (20)
002645' 000000  000240
002646' 001600  000000                          MWORD   <ADDR=16,CRTN>                  ; return to 13          (16)
002647' 000000  000240
002650' 001300  000000                          MWORD   <ADDR=13,CRTN>                  ; return to 10          (13)
002651' 000000  000240
002652' 777777  777777                          -1
```

```

                                      ;#*********************************************************************
                                      ;* Test 25 - Increment/Decrement Stack Pointer
                                      ;
                                      ; Description:  Verify that the 2910 stack pointer increments and
                                      ;               decrements correctly.  Do five nested subroutine
                                      ;               calls followed by four returns.
                                      ;
                                      ; Procedure:    Clear Port
                                      ;               Load test microcode
                                      ;               Set up RAR to start at location 1
                                      ;               Set MPRUN/SINGLE STEP (execute JZ)
                                      ;               Set MPRUN/SINGLE STEP 5 times (execute 5 CJS's)
                                      ;               Set MPRUN/SINGLE STEP 4 times (execute 4 CRTN's)
                                      ;               Read LAR, and verify address = 5
                                      ;
                                      ; Failure:      ---
                                      ;#*********************************************************************

                                      ; Test data

 002653' 254 00 0 00 002664'          TSTS25: JRST    TG25                  ; go start test
 002654' 240401  000025                       SEQ!MPROC!NDMP!ZSEQ!25        ; test mask
 002655' 002712' 010724'                      T25M,,[ASCIZ ^Increment/Decrement Stack Pointer^]
 002656' 010442' 010517'                      [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
 002657' 000000  002737'                      TSTS26                        ; failure test table
 002660' 000000  003032'                      TSTS30                        ;   ...
 002661' 000000  003122'                      TSTS31
 002662' 000000  003236'                      TSTS32
 002663' 777777  777777                       -1

                                      ; Start test

 002664' 201 00 0 00 000000'          TG25:   MOVEI   Z3                    ; get address of module start
 002665' 260 17 0 00 002603*                  GO      TRACE                 ; handle trace output
 002666' 201 01 0 00 002712'                  MOVEI   1,T25M                ; set up microcode address
 002667' 260 17 0 00 002605*                  GO      TLOAD                 ; load/verify it
 002670' 263 17 0 00 000000                   RTN                           ; failed - exit test

                                      ; Initialization

 002671' 260 17 0 00 002607*          TL25:   GO      IPACLR                ; clear port
 002672' 402 00 0 00 002610*                  SETZM   TSTSUB                ; initialize subtest number
 002673' 201 06 0 00 002706'                  MOVEI   6,TS25                ; get sstep table address

                                      ; Loop on single step table entries

 002674' 400 15 0 00 000000           TA25:   SETZ    ERFLG,                ; clear error flag
 002675' 260 17 0 00 002613*                  GO      SEXEC                 ; execute table entry
 002676' 254 00 0 00 002705'                  JRST    TX25                  ; end of sstep table
 002677' 254 00 0 00 002674'                  JRST    TA25                  ; keep looping after call
 002700' 474 15 0 00 000000                   SETO    ERFLG,                ; error - end address incorrect
```

```

                                      ; Handle error printouts and scope looping

002701' 027 00 0 00 002710'                   SCOPER  MA25                   ; print error message
002702' 254 00 0 00 002671'                   JRST    TL25                   ; loop on error
002703' 254 00 0 00 002705'                   JRST    TX25                   ; altmode exit
002704' 322 15 0 00 002674'                   JUMPE   ERFLG,TA25             ; do next sstep table entry

                                      ; End of test

002705' 263 17 0 00 000000            TX25:   RTN                            ; return

                                      ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

002706' 101200  010005                TS25:   STABLE  (SSSTRT,^D10,1,5)
002707' 000000  000000                        STABLE  (SSLAST)

                                      ; Error messages

002710' 140000  010733'               MA25:   MSG!TXNOT![ASCIZ /Stack pointer not working/]
002711' 000000000000#                         LAST!CALL!TXALL!SSPNT          ; print sequencer data

                                      ; Test microcode

002712' 000000  040000                T25M:   MWORD   <ADDR=0,CJS,J=4>               ; call subroutine       (0)
002713' 000000  000020
002714' 000100  000000                        MWORD   <JZ>                           ; initialize stack      (1)
002715' 000000  000000
002716' 000400  070000                        MWORD   <ADDR=4,CJS,J=7>               ; call subroutine       (4)
002717' 000000  000020
002720' 000700  120000                        MWORD   <ADDR=7,CJS,J=12>              ; call subroutine       (7)
002721' 000000  000020
002722' 001200  150000                        MWORD   <ADDR=12,CJS,J=15>             ; call subroutine       (12)
002723' 000000  000020
002724' 001500  200000                        MWORD   <ADDR=15,CJS,J=20>             ; call subroutine       (15)
002725' 000000  000020
002726' 002000  000000                        MWORD   <ADDR=20,CRTN>                 ; return to 16          (20)
002727' 000000  000240
002730' 001600  000000                        MWORD   <ADDR=16,CRTN>                 ; return to 13          (16)
002731' 000000  000240
002732' 001300  000000                        MWORD   <ADDR=13,CRTN>                 ; return to 10          (13)
002733' 000000  000240
002734' 001000  000000                        MWORD   <ADDR=10,CRTN>                 ; return to 5           (10)
002735' 000000  000240
002736' 777777  777777                        -1
```

```
                                        ;#********************************************************************
                                        ;* Test 26 - Increment/Decrement Stack Pointer
                                        ;
                                        ; Description:  Verify that the 2910 stack pointer increments and
                                        ;               decrements correctly.  Do five nested subroutine
                                        ;               calls followed by five returns.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode
                                        ;               Set up RAR to start at location 1
                                        ;               Set MPRUN/SINGLE STEP (execute JZ)
                                        ;               Set MPRUN/SINGLE STEP 5 times (execute 5 CJS's)
                                        ;               Set MPRUN/SINGLE STEP 5 times (execute 5 CRTN's)
                                        ;               Read LAR, and verify address = 2
                                        ;
                                        ; Failure:      ---
                                        ;#********************************************************************

                                        ; Test data

 002737' 254 00 0 00 002750'            TSTS26: JRST    TG26                    ; go start test
 002740' 240401  000026                         SEQ!MPROC!NDMP!ZSEQ!26          ; test mask
 002741' 002776' 010724'                        T26M,,[ASCIZ ^Increment/Decrement Stack Pointer^]
 002742' 010442' 010517'                        [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
 002743' 000000  003032'                        TSTS30                          ; failure test taole
 002744' 000000  003122'                        TSTS31                          ;   ...
 002745' 000000  003236'                        TSTS32
 002746' 000000  003314'                        TSTS33
 002747' 777777  777777                         -1

                                        ; Start test

 002750' 201 00 0 00 000000'            TG26:   MOVEI   Z3                      ; get address of module start
 002751' 260 17 0 00 002665*                    GO      TRACE                   ; handle trace output
 002752' 201 01 0 00 002776'                    MOVEI   1,T26M                  ; set up microcode address
 002753' 260 17 0 00 002667*                    GO      TLOAD                   ; load/verify it
 002754' 263 17 0 00 000000                     RTN                             ; failed - exit test

                                        ; Initialization

 002755' 260 17 0 00 002671*            TL26:   GO      IPACLR                  ; clear port
 002756' 402 00 0 00 002672*                    SETZM   TSTSUB                  ; initialize subtest number
 002757' 201 06 0 00 002772'                    MOVEI   6,TS26                  ; get sstep table address

                                        ; Loop on single step table entries

 002760' 400 15 0 00 000000             TA26:   SETZ    ERFLG,                  ; clear error flag
 002761' 260 17 0 00 002675*                    GO      SEXEC                   ; execute table entry
 002762' 254 00 0 00 002771'                    JRST    TX26                    ; end of sstep table
 002763' 254 00 0 00 002760'                    JRST    TA26                    ; keep looping after call
 002764' 474 15 0 00 000000                     SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

002765' 027 00 0 00 002774'                     SCOPER  MA26                    ; print error message
002766' 254 00 0 00 002755'                     JRST    TL26                    ; loop on error
002767' 254 00 0 00 002771'                     JRST    TX26                    ; altmode exit
002770' 322 15 0 00 002760'                     JUMPE   ERFLG,TA26              ; do next sstep table entry

                                        ; End of test

002771' 263 17 0 00 000000              TX26:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

002772' 101300  010001                  TS26:   STABLE  (SSSTRT,^D11,1,1)
002773' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

002774' 140000  010733'                 MA26:   MSG!TXNOT![ASCIZ /Stack pointer not working/]
002775' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

002776' 000000  040000                  T26M:   MWORD   <ADDR=0,CJS,J=4>                ; call subroutine       (0)
002777' 000000  000020
003000' 000100  000000                          MWORD   <JZ>                            ; initialize stack      (1)
003001' 000000  000000
003002' 000400  070000                          MWORD   <ADDR=4,CJS,J=7>                ; call subroutine       (4)
003003' 000000  000020
003004' 000700  120000                          MWORD   <ADDR=7,CJS,J=12>               ; call subroutine       (7)
003005' 000000  000020
003006' 001200  150000                          MWORD   <ADDR=12,CJS,J=15>              ; call subroutine       (12)
003007' 000000  000020
003010' 001500  200000                          MWORD   <ADDR=15,CJS,J=20>              ; call subroutine       (15)
003011' 000000  000020
003012' 002000  000000                          MWORD   <ADDR=20,CRTN>                  ; return to 16          (20)
003013' 000000  000240
003014' 001600  000000                          MWORD   <ADDR=16,CRTN>                  ; return to 13          (16)
003015' 000000  000240
003016' 001300  000000                          MWORD   <ADDR=13,CRTN>                  ; return to 10          (13)
003017' 000000  000240
003020' 001000  000000                          MWORD   <ADDR=10,CRTN>                  ; return to 5           (10)
003021' 000000  000240
003022' 000500  000000                          MWORD   <ADDR=5,CRTN>                   ; return to 1           (5)
003023' 000000  000240
003024' 777777  777777                          -1
```

```

                                    ;#*********************************************************************
                                    ;* Test 27 - x
                                    ;#*********************************************************************

003025' 263 17 0 00 000000          TSTS27: RTN                              ; return
003026' 000001  000027                      ZSEQ!27                          ; test mask
003027' 000000  010741'                     0,,[ASCIZ ^x^]
003030' 000000  000000                      0,,0
003031' 777777  777777                      -1
```

```
                                        ;#*********************************************************************
                                        ;* Test 30 - CJPP Instruction
                                        ;
                                        ; Description:  Verify that a CJPP instruction does a jump if the
                                        ;               TEST inputs are satisfied, and not if not.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode
                                        ;               Set up RAR to start at location 0
                                        ;               Set MPRUN/SINGLE STEP (execute CJPP,CJPP,CJPP,CJPP)
                                        ;               Read LAR, and verify address = 7
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

003032' 254 00 0 00 003043'             TSTS30: JRST    TG30                    ; go start test
003033' 240401  000030                          SEQ!MPROC!NDMP!ZSEQ!30          ; test mask
003034' 003071' 010742'                         T30M,,[ASCIZ ^CJPP Instruction^]
003035' 010442' 010517'                         [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
003036' 000000  003122'                         TSTS31                          ; failure test table
003037' 000000  003236'                         TSTS32                          ;   ...
003040' 000000  003314'                         TSTS33
003041' 000000  003360'                         TSTS34
003042' 777777  777777                          -1

                                        ; Start test

003043' 201 00 0 00 000000'             TG30:   MOVEI   Z3                      ; get address of module start
003044' 260 17 0 00 002751*                     GO      TRACE                   ; handle trace output
003045' 201 01 0 00 003071'                     MOVEI   1,T30M                  ; set up microcode address
003046' 260 17 0 00 002753*                     GO      TLOAD                   ; load/verify it
003047' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

003050' 260 17 0 00 002755*             TL30:   GO      IPACLR                  ; clear port
003051' 402 00 0 00 002756*                     SETZM   TSTSUB                  ; initialize subtest number
003052' 201 06 0 00 003065'                     MOVEI   6,TS30                  ; get sstep table address

                                        ; Loop on single step table entries

003053' 400 15 0 00 000000              TA30:   SETZ    ERFLG,                  ; clear error flag
003054' 260 17 0 00 002761*                     GO      SEXEC                   ; execute table entry
003055' 254 00 0 00 003064'                     JRST    TX30                    ; end of sstep table
003056' 254 00 0 00 003053'                     JRST    TA30                    ; keep looping after call
003057' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - end address incorrect
```

```
                                        ; Handle error printouts and scope looping

003060' 027 00 0 00 003067'                     SCOPER  MA30                    ; print error message
003061' 254 00 0 00 003050'                     JRST    TL30                    ; loop on error
003062' 254 00 0 00 003064'                     JRST    TX30                    ; altmode exit
003063' 322 15 0 00 003053'                     JUMPE   ERFLG,TA30              ; do next sstep table entry

                                        ; End of test

003064' 263 17 0 00 000000              TX30:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

003065' 100400  000007                  TS30:   STABLE  (SSSTRT,4,0,7)
003066' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

003067' 140000  010746'                 MA30:   MSG!TXNOT![ASCIZ /CJPP did not dispatch correctly/]
003070' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

003071' 000000  020000                  T30M:   MWORD   <ADDR=0,CJPP,J=2,DISA,CCOF>     ; disa/off      (0)
003072' 000000  010260
003073' 000100  100000                          MWORD   <JMAP,J=10>                     ;               (1)
003074' 000000  000040
003075' 000200  040000                          MWORD   <CJPP,J=4,DISA,CCON>            ; disa/on       (2)
003076' 000000  030260
003077' 000300  110000                          MWORD   <JMAP,J=11>                     ;               (3)
003100' 000000  000040
003101' 000400  060000                          MWORD   <CJPP,J=6,ENA,CCON>             ; ena/on        (4)
003102' 000400  030260
003103' 000500  120000                          MWORD   <JMAP,J=12>                     ;               (5)
003104' 000000  000040
003105' 000600  130000                          MWORD   <CJPP,J=13,ENA,CCOF>            ; ena/on        (6)
003106' 000400  010260
003107' 000700  070000                          MWORD   <JMAP,J=7>                      ;               (7)
003110' 000000  000040

003111' 001000  100000                          MWORD   <ADDR=10,JMAP,J=10>             ; error if here (10)
003112' 000000  000040
003113' 001100  110000                          MWORD   <ADDR=11,JMAP,J=11>             ; error if here (11)
003114' 000000  000040
003115' 001200  120000                          MWORD   <ADDR=12,JMAP,J=12>             ; error if here (12)
003116' 000000  000040
003117' 001300  130000                          MWORD   <ADDR=13,JMAP,J=13>             ; error if here (13)
003120' 000000  000040
003121' 777777  777777                          -1
```

```
                                       ;#*****************************************************************
                                       ;* Test 31 - CJPP Instruction
                                       ;
                                       ; Description:  Verify that the CJPP instruction pops  the  stack
                                       ;               if the TEST input is asserted.
                                       ;
                                       ; Procedure:    Clear Port
                                       ;               Load test microcode
                                       ;               Set up RAR to start at location 100
                                       ;               Set MPRUN/SINGLE STEP (execute JZ)
                                       ;               Set MPRUN/SINGLE STEP (execute CJS,CJS,CJPP,CRTN)
                                       ;               Set MPRUN/SINGLE STEP (execute CJS,CJS,CJPP,CRTN)
                                       ;               Set MPRUN/SINGLE STEP (execute CJS,CJS,CJPP,CRTN)
                                       ;               Read LAR, and verify address = 3
                                       ;
                                       ; Failure:      ---
                                       ;#*****************************************************************

                                       ; Test data

003122' 254 00 0 00 003133'            TSTS31: JRST    TG31                    ; go start test
003123' 240401  000031                         SEQ!MPROC!NDMP!ZSEQ!31          ; test mask
003124' 003161' 010742'                        T31M,,[ASCIZ ^CJPP Instruction^]
003125' 010442' 010517'                        [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
003126' 000000  003236'                        TSTS32                          ; failure test table
003127' 000000  003314'                        TSTS33                          ;   ...
003130' 000000  003360'                        TSTS34
003131' 000000  003444'                        TSTS35
003132' 777777  777777                         -1

                                       ; Start test

003133' 201 00 0 00 000000'            TG31:   MOVEI   Z3                      ; get address of module start
003134' 260 17 0 00 003044*                    GO      TRACE                   ; handle trace output
003135' 201 01 0 00 003161'                    MOVEI   1,T31M                  ; set up microcode address
003136' 260 17 0 00 003046*                    GO      TLOAD                   ; load/verify it
003137' 263 17 0 00 000000                     RTN                             ; failed - exit test

                                       ; Initialization

003140' 260 17 0 00 003050*            TL31:   GO      IPACLR                  ; clear port
003141' 402 00 0 00 003051*                    SETZM   TSTSUB                  ; initialize subtest number
003142' 201 06 0 00 003155'                    MOVEI   6,TS31                  ; get sstep table address

                                       ; Loop on single step table entries

003143' 400 15 0 00 000000             TA31:   SETZ    ERFLG,                  ; clear error flag
003144' 260 17 0 00 003054*                    GO      SEXEC                   ; execute table entry
003145' 254 00 0 00 003154'                    JRST    TX31                    ; end of sstep table
003146' 254 00 0 00 003143'                    JRST    TA31                    ; keep looping after call
003147' 474 15 0 00 000000                     SETO    ERFLG,                  ; error - end address incorrect
```

```
                                        ; Handle error printouts and scope looping

003150' 027 00 0 00 003157'                     SCOPER  MA31                    ; print error message
003151' 254 00 0 00 003140'                     JRST    TL31                    ; loop on error
003152' 254 00 0 00 003154'                     JRST    TX31                    ; altmode exit
003153' 322 15 0 00 003143'                     JUMPE   ERFLG,TA31              ; do next sstep table entry

                                        ; End of test

003154' 263 17 0 00 000000              TX31:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

003155' 101501  000003                  TS31:   STABLE  (SSSTRT,^D13,100,3)
003156' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

003157' 140000  010755'                 MA31:   MSG!TXNOT![ASCIZ /CJPP did not pop stack when test conditions met/]
003160' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

003161' 000000  100000                  T31M:   MWORD   <ADDR=0,CJS,J=10>                ; call subroutine      (0)
003162' 000000  000020
003163' 001000  200000                          MWORD   <ADDR=10,CJS,J=20>               ; call subroutine      (10)
003164' 000000  000020
003165' 001100  110000                          MWORD   <ADDR=11,JMAP,J=11>              ; error if here        (11)
003166' 000000  000040
003167' 002000  300000                          MWORD   <ADDR=20,CJPP,DISA,CCOF,J=30>    ; jump/pop (disa/off)  (20)
003170' 000000  010260
003171' 002100  210000                          MWORD   <ADDR=21,JMAP,J=21>              ; error if here        (21)
003172' 000000  000040
003173' 003000  000000                          MWORD   <ADDR=30,CRTN>                   ; return to 1          (30)
003174' 000000  000240
003175' 003100  310000                          MWORD   <ADDR=31,JMAP,J=31>              ; error if here        (31)
003176' 000000  000040

003177' 000100  120000                          MWORD   <ADDR=1,CJS,J=12>                ; call subroutine      (1)
003200' 000000  000020
003201' 001200  220000                          MWORD   <ADDR=12,CJS,J=22>               ; call subroutine      (12)
003202' 000000  000020
003203' 001300  130000                          MWORD   <ADDR=13,JMAP,J=13>              ; error if here        (13)
003204' 000000  000040
003205' 002200  320000                          MWORD   <ADDR=22,CJPP,DISA,CCON,J=32>    ; jump/pop (disa/on)   (22)
003206' 000000  030260
003207' 002300  230000                          MWORD   <ADDR=23,JMAP,J=23>              ; error if here        (23)
003210' 000000  000040
003211' 003200  000000                          MWORD   <ADDR=32,CRTN>                   ; return to 2          (32)
003212' 000000  000240
003213' 003300  330000                          MWORD   <ADDR=33,JMAP,J=33>              ; error if here        (33)
003214' 000000  000040

003215' 000200  140000                          MWORD   <ADDR=2,CJS,J=14>                ; call subroutine      (2)
```

```
2885  003216' 000000  000020
2886  003217' 001400  240000          MWORD   <ADDR=14,CJS,J=24>            ; call subroutine      (14)
2887  003220' 000000  000020
2888  003221' 001500  150000          MWORD   <ADDR=15,JMAP,J=15>           ; error if here        (15)
2889  003222' 000000  000040
2890  003223' 002400  340000          MWORD   <ADDR=24,CJPP,ENA,CCON,J=34>  ; jump and pop (ena/on) (24)
2891  003224' 000400  030260
2892  003225' 002500  250000          MWORD   <ADDR=25,JMAP,J=25>           ; error if here        (25)
2893  003226' 000000  000040
2894  003227' 003400  000000          MWORD   <ADDR=34,CRTN>                ; return to 2          (34)
2895  003230' 000000  000240
2896  003231' 003500  350000          MWORD   <ADDR=35,JMAP,J=35>           ; error if here        (35)
2897  003232' 000000  000040
2898
2899  003233' 010000  000000          MWORD   <ADDR=100,JZ>                 ; initialize stack     (100)
2900  003234' 000000  000000
2901  003235' 777777  777777          -1
```

```
                                   ;#*******************************************************************
                                   ;* Test 32 - CJPP Instruction
                                   ;
                                   ; Description:  Verify that the CJPP instruction does not pop the
                                   ;               stack if the TEST input is negated.
                                   ;
                                   ; Procedure:    Clear Port
                                   ;               Load test microcode
                                   ;               Set up RAR to start at location 1
                                   ;               Set MPRUN/SINGLE STEP (execute JZ,CJS,CJS)
                                   ;               Set MPRUN/SINGLE STEP (execute CJPP)
                                   ;               Set MPRUN/SINGLE STEP (execute CRTN)
                                   ;               Read LAR, and verify address = 5
                                   ;
                                   ; Failure:      ---
                                   ;#*******************************************************************

                                   ; Test data

003236' 254 00 0 00 003247'        TSTS32: JRST    TG32                    ; go start test
003237' 240401  000032                     SEQ!MPROC!NDMP!ZSEQ!32          ; test mask
003240' 003275' 010742'                    T32M,,[ASCIZ ^CJPP Instruction^]
003241' 010442' 010517'                    [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
003242' 000000  003314'                    TSTS33                          ; failure test table
003243' 000000  003360'                    TSTS34                          ;   ...
003244' 000000  003444'                    TSTS35
003245' 000000  003706'                    TSTS36
003246' 777777  777777                     -1

                                   ; Start test

003247' 201 00 0 00 000000'        TG32:   MOVEI   Z3                      ; get address of module start
003250' 260 17 0 00 003134*                GO      TRACE                   ; handle trace output
003251' 201 01 0 00 003275'                MOVEI   1,T32M                  ; set up microcode address
003252' 260 17 0 00 003136*                GO      TLOAD                   ; load/verify it
003253' 263 17 0 00 000000                 RTN                             ; failed - exit test

                                   ; Initialization

003254' 260 17 0 00 003140*        TL32:   GO      IPACLR                  ; clear port
003255' 402 00 0 00 003141*                SETZM   TSTSUB                  ; initialize subtest number
003256' 201 06 0 00 003271'                MOVEI   6,TS32                  ; get sstep table address

                                   ; Loop on single step table entries

003257' 400 15 0 00 000000         TA32:   SETZ    ERFLG,                  ; clear error flag
003260' 260 17 0 00 003144*                GO      SEXEC                   ; execute table entry
003261' 254 00 0 00 003270'                JRST    TX32                    ; end of sstep table
003262' 254 00 0 00 003257'                JRST    TA32                    ; keep looping after call
003263' 474 15 0 00 000000                 SETO    ERFLG,                  ; error - end address incorrect
```

```

                                ; Handle error printouts and scope looping

003264' 027 00 0 00 003273'             SCOPER  MA32                    ; print error message
003265' 254 00 0 00 003254'             JRST    TL32                    ; loop on error
003266' 254 00 0 00 003270'             JRST    TX32                    ; altmode exit
003267' 322 15 0 00 003257'             JUMPE   ERFLG,TA32              ; do next sstep table entry

                                ; End of test

003270' 263 17 0 00 000000      TX32:   RTN                             ; return

                                ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

003271' 100500  010005          TS32:   STABLE  (SSSTRT,5,1,5)
003272' 000000  000000                  STABLE  (SSLAST)

                                ; Error messages

003273' 140000  010767'         MA32:   MSG!TXNOT![ASCIZ /CJPP popped the stack when test conditions not met/]
003274' 000000000000#                   LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                ; Test microcode

003275' 000000  040000          T32M:   MWORD   <ADDR=0,CJS,J=4>                ; call subroutine       (0)
003276' 000000  000020
003277' 000100  000000                  MWORD   <ADDR=1,JZ>                     ; initialize stack      (1)
003300' 000000  000000
003301' 000400  070000                  MWORD   <ADDR=4,CJS,J=7>                ; call subroutine       (4)
003302' 000000  000020
003303' 000500  000000                  MWORD   <ADDR=5,JZ>                     ; initialize stack      (5)
003304' 000000  000000
003305' 000701  000000                  MWORD   <ADDR=7,CJPP,ENA,CCOF,J=100>    ; ena/off               (7)
003306' 000400  010260
003307' 001000  000000                  MWORD   <ADDR=10,CRTN>                  ; return to 5           (10)
003310' 000000  000240
003311' 010001  000000                  MWORD   <ADDR=100,JMAP,J=100>           ; error if here         (100)
003312' 000000  000040
003313' 777777  777777                  -1
```

```

                                    ;#*********************************************************************
                                    ;* Test 33 - LDCT Instruction
                                    ;
                                    ; Description:  Verify that the LDCT instruction does a  continue
                                    ;               among other things.
                                    ;
                                    ; Procedure:    Clear Port
                                    ;               Load test microcode and set RAR to 0
                                    ;               Set MPRUN/SINGLE STEP (execute JMAP,LDCT)
                                    ;               Read LAR, and verify address = 2
                                    ;
                                    ; Failure:      ---
                                    ;#*********************************************************************

                                    ; Test data

003314' 254 00 0 00 003325'         TSTS33: JRST    TG33                    ; go start test
003315' 240401  000033                      SEQ!MPROC!NDMP!ZSEQ!33          ; test mask
003316' 003353' 011002'                     T33M,,[ASCIZ ^LDCT Instruction^]
003317' 010442' 010517'                     [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
003320' 000000  003360'                     TSTS34                          ; failure test table
003321' 000000  003444'                     TSTS35                          ;   ...
003322' 000000  003706'                     TSTS36
003323' 000000  004000'                     TSTS37
003324' 777777  777777                      -1

                                    ; Start test

003325' 201 00 0 00 000000'         TG33:   MOVEI   Z3                      ; get address of module start
003326' 260 17 0 00 003250*                 GO      TRACE                   ; handle trace output
003327' 201 01 0 00 003353'                 MOVEI   1,T33M                  ; set up microcode address
003330' 260 17 0 00 003252*                 GO      TLOAD                   ; load/verify it
003331' 263 17 0 00 000000                  RTN                             ; failed - exit test

                                    ; Initialization

003332' 260 17 0 00 003254*         TL33:   GO      IPACLR                  ; clear port
003333' 402 00 0 00 003255*                 SETZM   TSTSUB                  ; initialize subtest number
003334' 201 06 0 00 003347'                 MOVEI   6,TS33                  ; get sstep table address

                                    ; Loop on single step table entries

003335' 400 15 0 00 000000          TA33:   SETZ    ERFLG,                  ; clear error flag
003336' 260 17 0 00 003260*                 GO      SEXEC                   ; execute table entry
003337' 254 00 0 00 003346'                 JRST    TX33                    ; end of sstep table
003340' 254 00 0 00 003335'                 JRST    TA33                    ; keep looping after call
003341' 474 15 0 00 000000                  SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

003342' 027 00 0 00 003351'                     SCOPER  MA33                    ; print error message
003343' 254 00 0 00 003332'                     JRST    TL33                    ; loop on error
003344' 254 00 0 00 003346'                     JRST    TX33                    ; altmode exit
003345' 322 15 0 00 003335'                     JUMPE   ERFLG,TA33              ; do next sstep table entry

                                        ; End of test

003346' 263 17 0 00 000000              TX33:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

003347' 100200  000002                  TS33:   STABLE  (SSSTRT,2,0,2)
003350' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

003351' 140000  011006'                 MA33:   MSG!TXNOT![ASCIZ /LDCT did not continue at PC+1/]
003352' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

003353' 000000  010000                  T33M:   MWORD   <ADDR=0,JMAP,J=1>
003354' 000000  000040
003355' 000100  770000                          MWORD   <LDCT,J=77>
003356' 000000  000300
003357' 777777  777777                          -1
```

```


                                   ;#*********************************************************************
                                   ;* Test 34 - LDCT Instruction
                                   ;
                                   ; Description:  Verify  that  the  LDCT  instruction  loads   the
                                   ;               register/counter with  the  data  on  the  direct
                                   ;               inputs.  At  the  same  time, verify that the JRP
                                   ;               instruction causes a jump to address specified in
                                   ;               the register/counter if the TEST  inputs  are not
                                   ;               satisfied.
                                   ;
                                   ;               Also, verify that the load is not affected by the
                                   ;               initial register/counter contents.
                                   ;
                                   ; Procedure:    Clear Port
                                   ;               Load test microcode
                                   ;               Set up RAR to start at location 100
                                   ;               Set MPRUN/SINGLE STEP (execute JZ)
                                   ;               Set MPRUN/SINGLE STEP (execute LDCT,JRP)
                                   ;               Set MPRUN/SINGLE STEP (execute LDCT,JRP)
                                   ;               Set MPRUN/SINGLE STEP (execute LDCT,JRP)
                                   ;               Read LAR, and verify address = 0
                                   ;
                                   ; Failure:      ---
                                   ;#*********************************************************************

                                   ; Test data

 003360' 254 00 0 00 003371'       TSTS34: JRST    TG34                    ; go start test
 003361' 240401  000034                    SEQ!MPROC!NDMP!ZSEQ!34          ; test mask
 003362' 003417' 011002'                   T34M,,[ASCIZ ^LDCT Instruction^]
 003363' 010442' 010517'                   [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
 003364' 000000  003444'                   TSTS35                          ; failure test table
 003365' 000000  003706'                   TSTS36                          ;   ...
 003366' 000000  004000'                   TSTS37
 003367' 000000  004076'                   TSTS40
 003370' 777777  777777                    -1

                                   ; Start test

 003371' 201 00 0 00 000000'       TG34:   MOVEI   Z3                      ; get address of module start
 003372' 260 17 0 00 003326*               GO      TRACE                   ; handle trace output
 003373' 201 01 0 00 003417'               MOVEI   1,T34M                  ; set up microcode address
 003374' 260 17 0 00 003330*               GO      TLOAD                   ; load/verify it
 003375' 263 17 0 00 000000                RTN                             ; failed - exit test

                                   ; Initialization

 003376' 260 17 0 00 003332*       TL34:   GO      IPACLR                  ; clear port
 003377' 402 00 0 00 003333*               SETZM   TSTSUB                  ; initialize subtest number
 003400' 201 06 0 00 003413'               MOVEI   6,TS34                  ; get sstep table address
```

```

                                     ; Loop on single step table entries

003401' 400 15 0 00 000000           TA34:   SETZ    ERFLG,                  ; clear error flag
003402' 260 17 0 00 003336*                  GO      SEXEC                   ; execute table entry
003403' 254 00 0 00 003412'                  JRST    TX34                    ; end of sstep table
003404' 254 00 0 00 003401'                  JRST    TA34                    ; keep looping after call
003405' 474 15 0 00 000000                   SETO    ERFLG,                  ; error - end address incorrect

                                     ; Handle error printouts and scope looping

003406' 027 00 0 00 003415'                  SCOPER  MA34                    ; print error message
003407' 254 00 0 00 003376'                  JRST    TL34                    ; loop on error
003410' 254 00 0 00 003412'                  JRST    TX34                    ; altmode exit
003411' 322 15 0 00 003401'                  JUMPE   ERFLG,TA34              ; do next sstep table entry

                                     ; End of test

003412' 263 17 0 00 000000           TX34:   RTN                             ; return

                                     ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

003413' 100701  000000               TS34:   STABLE  (SSSTRT,7,100,0)
003414' 000000  000000                       STABLE  (SSLAST)

                                     ; Error messages

003415' 140000  011014'              MA34:   MSG!TXNOT![ASCIZ /LDCT did not load cntr or JRP did not jump there/]
003416' 000000000000#                        LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                     ; Test microcode

003417' 010000  000000               T34M:   MWORD   <ADDR=100,JZ>                     ; init pc       (100)
003420' 000000  000000
003421' 000000  330000                       MWORD   <ADDR=0,LDCT,J=33>                ; load counter  (0)
003422' 000000  000300
003423' 000100  020000                       MWORD   <JRP,ENA,CCOF,J=2>                ; go to 33      (1)
003424' 000400  010160
003425' 000200  020000                       MWORD   <JMAP,J=2>                        ; error if here (2)
003426' 000000  000040
003427' 003300  440000                       MWORD   <ADDR=33,LDCT,J=44>               ; load counter  (33)
003430' 000000  000300
003431' 003400  350000                       MWORD   <JRP,ENA,CCOF,J=35>               ; go to 44      (34)
003432' 000400  010160
003433' 003500  350000                       MWORD   <JMAP,J=35>                       ; error if here (35)
003434' 000000  000040
003435' 004400  000000                       MWORD   <ADDR=44,LDCT,J=0>                ; load counter  (44)
003436' 000000  000300
003437' 004500  460000                       MWORD   <JRP,ENA,CCOF,J=46>               ; go to 0       (45)
003440' 000400  010160
003441' 004600  460000                       MWORD   <JMAP,J=46>                       ; error if here (46)
003442' 000000  000040
003443' 777777  777777                       -1
```

```

                                        ;#**************************************************************
                                        ;* Test 35 - Register/Counter
                                        ;
                                        ; Description:  Verify that the register/counter can hold various
                                        ;               data patterns.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode
                                        ;               Set up RAR to start at location 30
                                        ;               Set MPRUN/SINGLE STEP to execute all 21 LDCT,
                                        ;                 JRP pairs - total of 42 instructions
                                        ;               Read LAR, and verify address = 5776
                                        ;
                                        ;               This is done for register contents as follows:
                                        ;                 0,1,3,7,17,37,77,177,377,777,1777,3777,
                                        ;                 5777,5776,5775,5773,5767,5757,5737,5677,
                                        ;                 5577,5377,4777
                                        ;
                                        ; Failure:      ---
                                        ;#**************************************************************

                                        ; Test data

 003444' 254 00 0 00 003455'            TSTS35: JRST    TG35                    ; go start test
 003445' 240401  000035                         SEQ!MPROC!NDMP!ZSEQ!35          ; test mask
 003446' 003503' 011026'                        T35M,,[ASCIZ ^Register/Counter^]
 003447' 010442' 010517'                        [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
 003450' 000000  003706'                        TSTS36                          ; failure test table
 003451' 000000  004000'                        TSTS37                          ;   ...
 003452' 000000  004076'                        TSTS40
 003453' 000000  004176'                        TSTS41
 003454' 777777  777777                         -1

                                        ; Start test

 003455' 201 00 0 00 000000'            TG35:   MOVEI   Z3                      ; get address of module start
 003456' 260 17 0 00 003372*                    GO      TRACE                   ; handle trace output
 003457' 201 01 0 00 003503'                    MOVEI   1,T35M                  ; set up microcode address
 003460' 260 17 0 00 003374*                    GO      TLOAD                   ; load/verify it
 003461' 263 17 0 00 000000                     RTN                             ; failed - exit test

                                        ; Initialization

 003462' 260 17 0 00 003376*            TL35:   GO      IPACLR                  ; clear port
 003463' 402 00 0 00 003377*                    SETZM   TSTSUB                  ; initialize subtest number
 003464' 201 06 0 00 003477'                    MOVEI   6,TS35                  ; get sstep table address
```

```

                                     ; Loop on single step table entries

003465' 400 15 0 00 000000           TA35:   SETZ    ERFLG,                   ; clear error flag
003466' 260 17 0 00 003402*                  GO      SEXEC                    ; execute table entry
003467' 254 00 0 00 003476'                  JRST    TX35                     ; end of sstep table
003470' 254 00 0 00 003465'                  JRST    TA35                     ; keep looping after call
003471' 474 15 0 00 000000                   SETO    ERFLG,                   ; error - end address incorrect

                                     ; Handle error printouts and scope looping

003472' 027 00 0 00 003501'                  SCOPER  MA35                     ; print error message
003473' 254 00 0 00 003462'                  JRST    TL35                     ; loop on error
003474' 254 00 0 00 003476'                  JRST    TX35                     ; altmode exit
003475' 322 15 0 00 003465'                  JUMPE   ERFLG,TA35               ; do next sstep table entry

                                     ; End of test

003476' 263 17 0 00 000000           TX35:   RTN                              ; return

                                     ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

003477' 105300  275776               TS35:   STABLE  (SSSTRT,^D43,27,5776)
003500' 000000  000000                       STABLE  (SSLAST)

                                     ; Error messages

003501' 140000  011032'              MA35:   MSG!TXNOT![ASCIZ /Cntr not loaded properly or dispatch on it failed/]
003502' 000000000000#                        LAST!CALL!TXALL!SSPNT            ; print sequencer data

                                     ; Test microcode

003503' 002700  300000               T35M:   MWORD   <ADDR=27,JMAP,J=30>          ; init PC              (27)
003504' 000000  000040
003505' 003000  000000                       MWORD   <ADDR=30,LDCT,J=0>           ; cntr = 0             (30)
003506' 000000  000300
003507' 003100  440000                       MWORD   <JRP,ENA,CCOF,J=44>          ; go to cntr addr      (31)
003510' 000400  010160
003511' 003200  320000                       MWORD   <JMAP,J=32>                  ; error if here        (32)
003512' 000000  000040
003513' 000000  240000                       MWORD   <ADDR=0,JMAP,J=24>           ;                      (0)
003514' 000000  000040

003515' 002400  010000                       MWORD   <ADDR=24,LDCT,J=1>           ; cntr = 1             (24)
003516' 000000  000300
003517' 002500  440000                       MWORD   <JRP,ENA,CCOF,J=44>          ; go to cntr addr      (25)
003520' 000400  010160
003521' 002600  260000                       MWORD   <JMAP,J=26>                  ; error if here        (26)
003522' 000000  000040
003523' 000100  130000                       MWORD   <ADDR=1,JMAP,J=13>           ;                      (1)
003524' 000000  000040

003525' 001300  030000                       MWORD   <ADDR=13,LDCT,J=3>           ; cntr = 3             (13)
003526' 000000  000300
003527' 001400  440000                       MWORD   <JRP,ENA,CCOF,J=44>          ; go to cntr addr      (14)
```

```
003530' 000400  010160
003531' 001500  150000                    MWORD   <JMAP,J=15>                   ; error if here          (15)
003532' 000000  000040

003533' 000300  070000                    MWORD   <ADDR=3,LDCT,J=7>             ; cntr = 7               (3)
003534' 000000  000300
003535' 000400  440000                    MWORD   <JRP,ENA,CCOF,J=44>           ; go to cntr addr        (4)
003536' 000400  010160
003537' 000500  050000                    MWORD   <JMAP,J=5>                    ; error if here          (5)
003540' 000000  000040

003541' 000700  170000                    MWORD   <ADDR=7,LDCT,J=17>            ; cntr = 17              (7)
003542' 000000  000300
003543' 001000  440000                    MWORD   <JRP,ENA,CCOF,J=44>           ; go to cntr addr        (10)
003544' 000400  010160
003545' 001100  110000                    MWORD   <JMAP,J=11>                   ; error if here          (11)
003546' 000000  000040

003547' 001700  370000                    MWORD   <ADDR=17,LDCT,J=37>           ; cntr = 37              (17)
003550' 000000  000300
003551' 002000  440000                    MWORD   <JRP,ENA,CCOF,J=44>           ; go to cntr addr        (20)
003552' 000400  010160
003553' 002100  210000                    MWORD   <JMAP,J=21>                   ; error if here          (21)
003554' 000000  000040

003555' 003700  770000                    MWORD   <ADDR=37,LDCT,J=77>           ; cntr = 77              (37)
003556' 000000  000300
003557' 004000  440000                    MWORD   <JRP,ENA,CCOF,J=44>           ; go to cntr addr        (40)
003560' 000400  010160
003561' 004100  410000                    MWORD   <JMAP,J=41>                   ; error if here          (41)
003562' 000000  000040

003563' 007701  770000                    MWORD   <ADDR=77,LDCT,J=177>          ; cntr = 177             (77)
003564' 000000  000300
003565' 010000  440000                    MWORD   <JRP,ENA,CCOF,J=44>           ; go to cntr addr        (100)
003566' 000400  010160
003567' 010101  010000                    MWORD   <JMAP,J=101>                  ; error if here          (101)
003570' 000000  000040

003571' 017703  770000                    MWORD   <ADDR=177,LDCT,J=377>         ; cntr = 377             (177)
003572' 000000  000300
003573' 020000  440000                    MWORD   <JRP,ENA,CCOF,J=44>           ; go to cntr addr        (200)
003574' 000400  010160
003575' 020102  010000                    MWORD   <JMAP,J=201>                  ; error if here          (201)
003576' 000000  000040

003577' 037707  770000                    MWORD   <ADDR=377,LDCT,J=777>         ; cntr = 777             (377)
003600' 000000  000300
003601' 040000  440000                    MWORD   <JRP,ENA,CCOF,J=44>           ; go to cntr addr        (400)
003602' 000400  010160
003603' 040104  010000                    MWORD   <JMAP,J=401>                  ; error if here          (401)
003604' 000000  000040

003605' 077717  770000                    MWORD   <ADDR=777,LDCT,J=1777>        ; cntr = 1777            (777)
003606' 000000  000300
```

```
003607' 100000  440000          MWORD   <JRP,ENA,CCOF,J=44>             ; go to cntr addr       (1000)
003610' 000400  010160
003611' 100110  010000          MWORD   <JMAP,J=1001>                   ; error if here         (1001)
003612' 000000  000040

003613' 177737  770000          MWORD   <ADDR=1777,LDCT,J=3777>         ; cntr = 3777           (1777)
003614' 000000  000300
003615' 200000  440000          MWORD   <JRP,ENA,CCOF,J=44>             ; go to cntr addr       (2000)
003616' 000400  010160
003617' 200120  010000          MWORD   <JMAP,J=2001>                   ; error if here         (2001)
003620' 000000  000040

003621' 377747  770000          MWORD   <ADDR=3777,LDCT,J=4777>         ; cntr = 4777           (3777)
003622' 000000  000300
003623' 400000  440000          MWORD   <JRP,ENA,CCOF,J=44>             ; go to cntr addr       (4000)
003624' 000400  010160
003625' 400140  010000          MWORD   <JMAP,J=4001>                   ; error if here         (4001)
003626' 000000  000040

003627' 477753  770000          MWORD   <ADDR=4777,LDCT,J=5377>         ; cntr = 5377           (4777)
003630' 000000  000300
003631' 500000  440000          MWORD   <JRP,ENA,CCOF,J=44>             ; go to cntr addr       (5000)
003632' 000400  010160
003633' 500150  010000          MWORD   <JMAP,J=5001>                   ; error if here         (5001)
003634' 000000  000040

003635' 537755  770000          MWORD   <ADDR=5377,LDCT,J=5577>         ; cntr = 5577           (5377)
003636' 000000  000300
003637' 540000  440000          MWORD   <JRP,ENA,CCOF,J=44>             ; go to cntr addr       (5400)
003640' 000400  010160
003641' 540154  010000          MWORD   <JMAP,J=5401>                   ; error if here         (5401)
003642' 000000  000040

003643' 557756  770000          MWORD   <ADDR=5577,LDCT,J=5677>         ; cntr = 5677           (5577)
003644' 000000  000300
003645' 560000  440000          MWORD   <JRP,ENA,CCOF,J=44>             ; go to cntr addr       (5600)
003646' 000400  010160
003647' 560156  010000          MWORD   <JMAP,J=5601>                   ; error if here         (5601)
003650' 000000  000040

003651' 567757  370000          MWORD   <ADDR=5677,LDCT,J=5737>         ; cntr = 5737           (5677)
003652' 000000  000300
003653' 570000  440000          MWORD   <JRP,ENA,CCOF,J=44>             ; go to cntr addr       (5700)
003654' 000400  010160
003655' 570157  010000          MWORD   <JMAP,J=5701>                   ; error if here         (5701)
003656' 000000  000040

003657' 573757  570000          MWORD   <ADDR=5737,LDCT,J=5757>         ; cntr = 5757           (5737)
003660' 000000  000300
003661' 574000  440000          MWORD   <JRP,ENA,CCOF,J=44>             ; go to cntr addr       (5740)
003662' 000400  010160
003663' 574157  510000          MWORD   <JMAP,J=5751>                   ; error if here         (5741)
003664' 000000  000040

003665' 575757  730000          MWORD   <ADDR=5757,LDCT,J=5773>         ; cntr = 5773           (5757)
```

```
003666' 000000  000340
003667' 576000  440000          MWORD   <JRP,ENA,CCOF,J=44>             ; go to cntr addr       (5760)
003670' 000400  010160
003671' 576157  610000          MWORD   <JMAP,J=5761>                   ; error if here         (5761)
003672' 000000  000040

003673' 577357  760000          MWORD   <ADDR=5773,LDCT,J=5776>         ; cntr = 5775           (5773)
003674' 000000  000300
003675' 577400  440000          MWORD   <JRP,ENA,CCOF,J=44>             ; go to cntr addr       (5774)
003676' 000400  010160
003677' 577557  750000          MWORD   <JMAP,J=5775>                   ; error if here         (5775)
003700' 000000  000040

003701' 577600  000000          MWORD   <CONT>                          ; done - ok             (5776)
003702' 000000  000340
003703' 577757  770000          MWORD   <JMAP,J=5777>                   ; error if here         (5777)
003704' 000000  000040
003705' 777777  777777          -1
```

```

                                 ;#*********************************************************************
                                 ;* Test 36 - JRP Instruction
                                 ;
                                 ; Description:  Verify that a JRP jumps to  address  specified at
                                 ;               direct  inputs  if TEST inputs are satisfied, and
                                 ;               not if not.  Also, that register/counter contents
                                 ;               are not affected by the JRP.
                                 ;
                                 ; Procedure:    Clear Port
                                 ;               Load test microcode
                                 ;               Set up RAR to start at location 100
                                 ;               Set MPRUN/SINGLE STEP (execute JZ)
                                 ;               Set MPRUN/SINGLE STEP (execute LDCT,JRP,JRP,JRP)
                                 ;               Set MPRUN/SINGLE STEP (execute JRP,JRP)
                                 ;               Read LAR, and verify address = 11
                                 ;
                                 ; Failure:      ---
                                 ;#*********************************************************************

                                 ; Test data

003706' 254 00 0 00 003717'      TSTS36: JRST    TG36                    ; go start test
003707' 240401  000036                   SEQ!MPROC!NDMP!ZSEQ!36          ; test mask
003710' 003745' 011044'                  T36M,,[ASCIZ ^JRP Instruction^]
003711' 010442' 010517'                  [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
003712' 000000  004000'                  TSTS37                          ; failure test table
003713' 000000  004076'                  TSTS40                          ;   ...
003714' 000000  004176'                  TSTS41
003715' 000000  004256'                  TSTS42
003716' 777777  777777                   -1

                                 ; Start test

003717' 201 00 0 00 000000'      TG36:   MOVEI   Z3                      ; get address of module start
003720' 260 17 0 00 003456*              GO      TRACE                   ; handle trace output
003721' 201 01 0 00 003745'              MOVEI   1,T36M                  ; set up microcode address
003722' 260 17 0 00 003460*              GO      TLOAD                   ; load/verify it
003723' 263 17 0 00 000000               RTN                             ; failed - exit test

                                 ; Initialization

003724' 260 17 0 00 003462*      TL36:   GO      IPACLR                  ; clear port
003725' 402 00 0 00 003463*              SETZM   TSTSUB                  ; initialize subtest number
003726' 201 06 0 00 003741'              MOVEI   6,TS36                  ; get sstep table address

                                 ; Loop on single step table entries

003727' 400 15 0 00 000000       TA36:   SETZ    ERFLG,                  ; clear error flag
003730' 260 17 0 00 003466*              GO      SEXEC                   ; execute table entry
003731' 254 00 0 00 003740'              JRST    TX36                    ; end of sstep table
003732' 254 00 0 00 003727'              JRST    TA36                    ; keep looping after call
003733' 474 15 0 00 000000               SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

003734' 027 00 0 00 003743'                     SCOPER  MA36                    ; print error message
003735' 254 00 0 00 003724'                     JRST    TL36                    ; loop on error
003736' 254 00 0 00 003740'                     JRST    TX36                    ; altmode exit
003737' 322 15 0 00 003727'                     JUMPE   ERFLG,TA36              ; do next sstep table entry

                                        ; End of test

003740' 263 17 0 00 000000              TX36:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

003741' 101201  000011                  TS36:   STABLE  (SSSTRT,^D10,100,11)
003742' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

003743' 140000  011050'                 MA36:   MSG!TXNOT![ASCIZ /JRP did not dispatch correctly or cntr messed up/]
003744' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

003745' 000000  110000                  T36M:   MWORD   <ADDR=0,LDCT,J=11>              ; load counter          (0)
003746' 000000  000300
003747' 000100  030000                          MWORD   <JRP,DISA,CCOF,J=3>             ; go to 3 (disa/off)    (1)
003750' 000000  010160
003751' 000200  020000                          MWORD   <JMAP,J=2>                      ; error if here         (2)
003752' 000000  000040
003753' 000300  050000                          MWORD   <JRP,DISA,CCON,J=5>             ; go to 5 (disa/on)     (3)
003754' 000000  030160
003755' 000400  040000                          MWORD   <JMAP,J=4>                      ; error if here         (4)
003756' 000000  000040
003757' 000500  070000                          MWORD   <JRP,ENA,CCON,J=7>              ; go to 7 (ena/on)      (5)
003760' 000400  030160
003761' 000600  060000                          MWORD   <JMAP,J=6>                      ; error if here         (6)
003762' 000000  000040
003763' 000700  770000                          MWORD   <JRP,ENA,CCOF,J=77>             ; go to cntr (ena/off)  (7)
003764' 000400  010160
003765' 001000  100000                          MWORD   <JMAP,J=10>                     ; error if here         (10)
003766' 000000  000040
003767' 001100  770000                          MWORD   <JRP,ENA,CCOF,J=77>             ; go to cntr (ena/off)  (11)
003770' 000400  010160
003771' 001200  120000                          MWORD   <JMAP,J=12>                     ; error if here         (12)
003772' 000000  000040

003773' 007700  770000                          MWORD   <ADDR=77,JMAP,J=77>             ; error if here         (77)
003774' 000000  000040
003775' 010000  000000                          MWORD   <ADDR=100,JZ>                   ; init PC               (100)
003776' 000000  000000
003777' 777777  777777                          -1
```

```
                                            ;#*********************************************************************
                                            ;* Test 37 - JSRP Instruction
                                            ;
                                            ; Description:   Verify that a JSRP jumps to  address specified in
                                            ;                the register/counter if the TEST  inputs  are not
                                            ;                satisfied.
                                            ;
                                            ;                Also, verify that JSRP jumps to D inputs  if TEST
                                            ;                inputs are satisfied.
                                            ;
                                            ; Procedure:     Clear Port
                                            ;                Load test microcode
                                            ;                Set up RAR to start at location 100
                                            ;                Set MPRUN/SINGLE STEP (execute JZ)
                                            ;                Set MPRUN/SINGLE STEP (execute LDCT,JSRP)
                                            ;                Set MPRUN/SINGLE STEP (execute LDCT,JSRP)
                                            ;                Set MPRUN/SINGLE STEP (execute LDCT,JSRP)
                                            ;                Set MPRUN/SINGLE STEP (execute LDCT,JSRP)
                                            ;                Read LAR, and verify address = 14
                                            ;
                                            ; Failure:       ---
                                            ;#*********************************************************************

                                            ; Test data

 004000' 254 00 0 00 004011'                TSTS37: JRST    TG37                            ; go start test
 004001' 240401  000037                             SEQ!MPROC!NDMP!ZSEQ!37                  ; test mask
 004002' 004037' 011062'                            T37M,,[ASCIZ ^JSRP Instruction^]
 004003' 010442' 010517'                            [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
 004004' 000000  004076'                            TSTS40                                  ; failure test table
 004005' 000000  004176'                            TSTS41                                  ;   ...
 004006' 000000  004256'                            TSTS42
 004007' 000000  004355'                            TSTS43
 004010' 777777  777777                             -1

                                            ; Start test

 004011' 201 00 0 00 000000'                TG37:   MOVEI   Z3                              ; get address of module start
 004012' 260 17 0 00 003720*                        GO      TRACE                           ; handle trace output
 004013' 201 01 0 00 004037'                        MOVEI   1,T37M                          ; set up microcode address
 004014' 260 17 0 00 003722*                        GO      TLOAD                           ; load/verify it
 004015' 263 17 0 00 000000                         RTN                                     ; failed - exit test
```

```

                                         ; Initialization

004016' 260 17 0 00 003724*              TL37:   GO      IPACLR                    ; clear port
004017' 402 00 0 00 003725*                      SETZM   TSTSUB                    ; initialize subtest number
004020' 201 06 0 00 004033'                      MOVEI   6,TS37                    ; get sstep table address

                                         ; Loop on single step table entries

004021' 400 15 0 00 000000               TA37:   SETZ    ERFLG,                    ; clear error flag
004022' 260 17 0 00 003730*                      GO      SEXEC                     ; execute table entry
004023' 254 00 0 00 004032'                      JRST    TX37                      ; end of sstep table
004024' 254 00 0 00 004021'                      JRST    TA37                      ; keep looping after call
004025' 474 15 0 00 000000                       SETO    ERFLG,                    ; error - end address incorrect

                                         ; Handle error printouts and scope looping

004026' 027 00 0 00 004035'                      SCOPER  MA37                      ; print error message
004027' 254 00 0 00 004016'                      JRST    TL37                      ; loop on error
004030' 254 00 0 00 004032'                      JRST    TX37                      ; altmode exit
004031' 322 15 0 00 004021'                      JUMPE   ERFLG,TA37                ; do next sstep table entry

                                         ; End of test

004032' 263 17 0 00 000000               TX37:   RTN                               ; return

                                         ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

004033' 101101  000014                   TS37:   STABLE  (SSSTRT,^D9,100,14)
004034' 000000  000000                           STABLE  (SSLAST)

                                         ; Error messages

004035' 140000  011066'                  MA37:   MSG!TXNOT![ASCIZ /JSRP did not dispatch correctly/]
004036' 000000000000#                            LAST!CALL!TXALL!SSPNT             ; print sequencer data
```

```
3586
3587                                     ; Test microcode
3588
3589  004037' 010000  000000             T37M:   MWORD   <ADDR=100,JZ>                 ; init PC       (100)
3590  004040' 000000  000000
3591  004041' 000000  020000                     MWORD   <ADDR=0,LDCT,J=2>             ; load counter  (0)
3592  004042' 000000  000300
3593  004043' 000100  030000                     MWORD   <JSRP,DISA,CCOF,J=3>          ; disa/off      (1)
3594  004044' 000000  010120
3595  004045' 000200  020000                     MWORD   <JMAP,J=2>                    ; error if here (2)
3596  004046' 000000  000040
3597
3598  004047' 000300  050000                     MWORD   <LDCT,J=5>                    ; load counter  (3)
3599  004050' 000000  000300
3600  004051' 000400  060000                     MWORD   <JSRP,DISA,CCON,J=6>          ; disa/on       (4)
3601  004052' 000000  030120
3602  004053' 000500  050000                     MWORD   <JMAP,J=5>                    ; error if here (5)
3603  004054' 000000  000040
3604
3605  004055' 000600  100000                     MWORD   <LDCT,J=10>                   ; load counter  (6)
3606  004056' 000000  000300
3607  004057' 000700  110000                     MWORD   <JSRP,ENA,CCON,J=11>          ; ena/on        (7)
3608  004060' 000400  030120
3609  004061' 001000  120000                     MWORD   <JMAP,J=12>                   ; error if here (10)
3610  004062' 000000  000040
3611
3612  004063' 001100  140000                     MWORD   <LDCT,J=14>                   ; load counter  (11)
3613  004064' 000000  000300
3614  004065' 001200  130000                     MWORD   <JSRP,ENA,CCOF,J=13>          ; ena/off       (12)
3615  004066' 000400  010120
3616  004067' 001300  130000                     MWORD   <JMAP,J=13>                   ; error if here (13)
3617  004070' 000000  000040
3618
3619  004071' 001400  000000                     MWORD   <CONT>                        ; done - ok     (14)
3620  004072' 000000  000340
3621  004073' 001500  150000                     MWORD   <JMAP,J=15>                   ; error if here (15)
3622  004074' 000000  000040
3623  004075' 777777  777777                     -1
```

```
                                        ;#**********************************************************************
                                        ;* Test 40 - JSRP Instruction
                                        ;
                                        ; Description:  Verify that  the  JSRP  pushes 'CRAM address + 1'
                                        ;               onto the stack regardless of TEST input.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode
                                        ;               Set up RAR to start at location 100
                                        ;               Set MPRUN/SINGLE STEP (execute JZ,PUSH)
                                        ;               Set MPRUN/SINGLE STEP (execute LDCT,JSRP)
                                        ;               Set MPRUN/SINGLE STEP (execute LDCT,JSRP)
                                        ;               Set MPRUN/SINGLE STEP (execute LDCT,JSRP)
                                        ;               Set MPRUN/SINGLE STEP (execute LDCT,JSRP)
                                        ;               Set MPRUN/SINGLE STEP (execute CRTN,CRTN,CRTN,CRTN,CRTN)
                                        ;               Read LAR, and verify address = 1
                                        ;
                                        ; Failure:      ---
                                        ;#**********************************************************************

                                        ; Test data

004076' 254 00 0 00 004107'             TSTS40: JRST    TG40                    ; go start test
004077' 240401  000040                          SEQ!MPROC!NDMP!ZSEQ!40          ; test mask
004100' 004135' 011062'                         T40M,,[ASCIZ ^JSRP Instruction^]
004101' 010442' 010517'                         [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
004102' 000000  004176'                         TSTS41                          ; failure test table
004103' 000000  004256'                         TSTS42                          ;   ...
004104' 000000  004355'                         TSTS43
004105' 000000  004447'                         TSTS44
004106' 777777  777777                          -1

                                        ; Start test

004107' 201 00 0 00 000000'             TG40:   MOVEI   Z3                      ; get address of module start
004110' 260 17 0 00 004012*                     GO      TRACE                   ; handle trace output
004111' 201 01 0 00 004135'                     MOVEI   1,T40M                  ; set up microcode address
004112' 260 17 0 00 004014*                     GO      TLOAD                   ; load/verify it
004113' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

004114' 260 17 0 00 004016*             TL40:   GO      IPACLR                  ; clear port
004115' 402 00 0 00 004017*                     SETZM   TSTSUB                  ; initialize subtest number
004116' 201 06 0 00 004131'                     MOVEI   6,TS40                  ; get sstep table address
```

```

                                  ; Loop on single step table entries

004117' 400 15 0 00 000000        TA40:   SETZ    ERFLG,                  ; clear error flag
004120' 260 17 0 00 004022*               GO      SEXEC                   ; execute table entry
004121' 254 00 0 00 004130'               JRST    TX40                    ; end of sstep table
004122' 254 00 0 00 004117'               JRST    TA40                    ; keep looping after call
004123' 474 15 0 00 000000                SETO    ERFLG,                  ; error - end address incorrect

                                  ; Handle error printouts and scope looping

004124' 027 00 0 00 004133'               SCOPER  MA40                    ; print error message
004125' 254 00 0 00 004114'               JRST    TL40                    ; loop on error
004126' 254 00 0 00 004130'               JRST    TX40                    ; altmode exit
004127' 322 15 0 00 004117'               JUMPE   ERFLG,TA40              ; do next sstep table entry

                                  ; End of test

004130' 263 17 0 00 000000        TX40:   RTN                             ; return

                                  ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

004131' 101701  000001            TS40:   STABLE  (SSSTRT,^D15,100,1)
004132' 000000  000000                    STABLE  (SSLAST)

                                  ; Error messages

004133' 140000  011075'           MA40:   MSG!TXNOT![ASCIZ /JSRP did not do a push/]
004134' 000000000000#                     LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                  ; Test microcode

004135' 000000  000000            T40M:   MWORD   <ADDR=0,PUSH>                   ; push 1 (#1)           (0)
004136' 000000  000100

004137' 000100  770000                    MWORD   <LDCT,J=77>                     ; load counter          (1)
004140' 000000  000300
004141' 000200  040000                    MWORD   <JSRP,DISA,CCOF,J=4>            ; go to 4 (push #2)     (2)
004142' 000000  010120
004143' 000300  000000                    MWORD   <CRTN>                          ; return to 1 (pop #5)  (3)
004144' 000000  000240

004145' 000400  770000                    MWORD   <LDCT,J=77>                     ; load counter          (4)
004146' 000000  000300
004147' 000500  070000                    MWORD   <JSRP,DISA,CCON,J=7>            ; go to 7 (push #3)     (5)
004150' 000000  030120
004151' 000600  000000                    MWORD   <CRTN>                          ; return to 3 (pop #4)  (6)
004152' 000000  000240

004153' 000700  770000                    MWORD   <LDCT,J=77>                     ; load counter          (7)
004154' 000000  000300
004155' 001000  120000                    MWORD   <JSRP,ENA,CCON,J=12>            ; go to 12 (push #4)    (10)
004156' 000400  030120
004157' 001100  000000                    MWORD   <CRTN>                          ; return to 6 (pop #3)  (11)
004160' 000000  000240
```

```

004161' 001200  150000           MWORD    <LDCT,J=15>                  ; load counter           (12)
004162' 000000  000300
004163' 001300  770000           MWORD    <JSRP,ENA,CCOF,J=77>         ; go to cnt adr (push #5) (13)
004164' 000400  010120
004165' 001400  000000           MWORD    <CRTN>                       ; return to 11 (pop #2)   (14)
004166' 000000  000240

004167' 001500  000000           MWORD    <CRTN>                       ; return to 14 (pop #1)   (15)
004170' 000000  000240
004171' 007700  770000           MWORD    <ADDR=77,JMAP,J=77>          ; error if here           (77)
004172' 000000  000040

004173' 010000  000000           MWORD    <ADDR=100,JZ>                ; initialize stack        (100)
004174' 000000  000000
004175' 777777  777777           -1
```

```

                                  ;#****************************************************************
                                  ;* Test 41 - JSRP Instruction
                                  ;
                                  ; Description:  Verify that the JSRP does  not  affect  register/
                                  ;               counter regardless of TEST inputs.
                                  ;
                                  ; Procedure:    Clear Port
                                  ;               Load test microcode
                                  ;               Set MPRUN/SINGLE STEP (execute JZ)
                                  ;               Set MPRUN/SINGLE STEP (execute LDCT,JSRP,JSRP,JSRP,JSRP)
                                  ;               Read LAR, and verify address = 6
                                  ;
                                  ; Failure:      ---
                                  ;#****************************************************************

                                  ; Test data

004176' 254 00 0 00 004207'       TSTS41: JRST    TG41                    ; go start test
004177' 240401  000041                    SEQ!MPROC!NDMP!ZSEQ!41          ; test mask
004200' 004235' 011062'                   T41M,,[ASCIZ ^JSRP Instruction^]
004201' 010442' 010517'                   [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
004202' 000000  004256'                   TSTS42                          ; failure test table
004203' 000000  004355'                   TSTS43                          ;   ...
004204' 000000  004447'                   TSTS44
004205' 000000  004527'                   TSTS45
004206' 777777  777777                    -1

                                  ; Start test

004207' 201 00 0 00 000000'       TG41:   MOVEI   Z3                      ; get address of module start
004210' 260 17 0 00 004110*               GO      TRACE                   ; handle trace output
004211' 201 01 0 00 004235'               MOVEI   1,T41M                  ; set up microcode address
004212' 260 17 0 00 004112*               GO      TLOAD                   ; load/verify it
004213' 263 17 0 00 000000                RTN                             ; failed - exit test

                                  ; Initialization

004214' 260 17 0 00 004114*       TL41:   GO      IPACLR                  ; clear port
004215' 402 00 0 00 004115*               SETZM   TSTSUB                  ; initialize subtest number
004216' 201 06 0 00 004231'               MOVEI   6,TS41                  ; get sstep table address

                                  ; Loop on single step table entries

004217' 400 15 0 00 000000        TA41:   SETZ    ERFLG,                  ; clear error flag
004220' 260 17 0 00 004120*               GO      SEXEC                   ; execute table entry
004221' 254 00 0 00 004230'               JRST    TX41                    ; end of sstep table
004222' 254 00 0 00 004217'               JRST    TA41                    ; keep looping after call
004223' 474 15 0 00 000000                SETO    ERFLG,                  ; error - end address incorrect
```

```

                                            ; Handle error printouts and scope looping

004224' 027 00 0 00 004233'                          SCOPER  MA41                   ; print error message
004225' 254 00 0 00 004214'                          JRST    TL41                   ; loop on error
004226' 254 00 0 00 004230'                          JRST    TX41                   ; altmode exit
004227' 322 15 0 00 004217'                          JUMPE   ERFLG,TA41             ; do next sstep table entry

                                            ; End of test

004230' 263 17 0 00 000000          TX41:    RTN                            ; return

                                            ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

004231' 100601  000006              TS41:    STABLE  (SSSTRT,6,100,6)
004232' 000000  000000                       STABLE  (SSLAST)

                                            ; Error messages

004233' 140000  011102'             MA41:    MSG!TXNOT![ASCIZ /JSRP affected cntr and it shouldn't have/]
004234' 000000000000#                        LAST!CALL!TXALL!SSPNT          ; print sequencer data

                                            ; Test microcode

004235' 010000  000000              T41M:    MWORD   <ADDR=100,JZ>                  ; init PC       (100)
004236' 000000  000000
004237' 000000  060000                       MWORD   <ADDR=0,LDCT,J=6>              ; load counter  (0)
004240' 000000  000300
004241' 000100  020000                       MWORD   <JSRP,DISA,CCOF,J=2>           ; disa/off      (1)
004242' 000000  010120
004243' 000200  030000                       MWORD   <JSRP,DISA,CCON,J=3>           ; disa/on       (2)
004244' 000000  030120
004245' 000300  040000                       MWORD   <JSRP,ENA,CCON,J=4>            ; ena/on        (3)
004246' 000400  030120
004247' 000400  050000                       MWORD   <JSRP,ENA,CCOF,J=5>            ; ena/off       (4)
004250' 000400  010120
004251' 000500  050000                       MWORD   <JMAP,J=5>                     ; error if here (5)
004252' 000000  000040
004253' 000600  050000                       MWORD   <JMAP,J=5>                     ; here if ok    (6)
004254' 000000  000040
004255' 777777  777777                       -1
```

```
                                                ;#*******************************************************************
                                                ;* Test 42 - PUSH Instruction
                                                ;
                                                ; Description:  Verify that the  PUSH  instruction  pushes  'CRAM
                                                ;               address + 1' onto the stack.
                                                ;
                                                ; Procedure:    Clear Port
                                                ;               Load test microcode
                                                ;
                                                ;               Set up RAR to start at location 4
                                                ;               Set MPRUN/SINGLE STEP (execute JZ,CJS,PUSH,CRTN)
                                                ;               Read LAR, and verify address = 6
                                                ;
                                                ;               Set up RAR to start at location 1
                                                ;               Set MPRUN/SINGLE STEP (execute CJS,PUSH,CRTN)
                                                ;               Read LAR, and verify address = 10
                                                ;
                                                ;               Set up RAR to start at location 2
                                                ;               Set MPRUN/SINGLE STEP (execute CJS,PUSH,CRTN)
                                                ;               Read LAR, and verify address = 12
                                                ;
                                                ;               Set up RAR to start at location 3
                                                ;               Set MPRUN/SINGLE STEP (execute CJS,PUSH,CRTN)
                                                ;               Read LAR, and verify address = 14
                                                ;
                                                ; Failure:      ---
                                                ;#*******************************************************************

                                                ; Test data

004256' 254 00 0 00 004267'                     TSTS42: JRST    TG42                    ; go start test
004257' 240401  000042                                  SEQ!MPROC!NDMP!ZSEQ!42          ; test mask
004260' 004320' 011113'                                 T42M,,[ASCIZ ^PUSH Instruction^]
004261' 010442' 010517'                                 [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
004262' 000000  004355'                                 TSTS43                          ; failure test table
004263' 000000  004447'                                 TSTS44                          ;   ...
004264' 000000  004527'                                 TSTS45
004265' 000000  004616'                                 TSTS46
004266' 777777  777777                                  -1

                                                ; Start test

004267' 201 00 0 00 000000'                     TG42:   MOVEI   Z3                      ; get address of module start
004270' 260 17 0 00 004210*                             GO      TRACE                   ; handle trace output
004271' 201 01 0 00 004320'                             MOVEI   1,T42M                  ; set up microcode address
004272' 260 17 0 00 004212*                             GO      TLOAD                   ; load/verify it
004273' 263 17 0 00 000000                              RTN                             ; failed - exit test
```

```

                                        ; Initialization

004274' 260 17 0 00 004214*             TL42:   GO      IPACLR                  ; clear port
004275' 402 00 0 00 004215*                     SETZM   TSTSUB                  ; initialize subtest number
004276' 201 06 0 00 004311'                     MOVEI   6,TS42                  ; get sstep table address

                                        ; Loop on single step table entries

004277' 400 15 0 00 000000              TA42:   SETZ    ERFLG,                  ; clear error flag
004300' 260 17 0 00 004220*                     GO      SEXEC                   ; execute table entry
004301' 254 00 0 00 004310'                     JRST    TX42                    ; end of sstep table
004302' 254 00 0 00 004277'                     JRST    TA42                    ; keep looping after call
004303' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - end address incorrect

                                        ; Handle error printouts and scope looping

004304' 027 00 0 00 004316'                     SCOPER  MA42                    ; print error message
004305' 254 00 0 00 004274'                     JRST    TL42                    ; loop on error
004306' 254 00 0 00 004310'                     JRST    TX42                    ; altmode exit
004307' 322 15 0 00 004277'                     JUMPE   ERFLG,TA42              ; do next sstep table entry

                                        ; End of test

004310' 263 17 0 00 000000              TX42:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

004311' 100400  040006                  TS42:   STABLE  (SSSTRT,4,4,6)
004312' 100300  010010                          STABLE  (SSSTRT,3,1,10)
004313' 100300  020012                          STABLE  (SSSTRT,3,2,12)
004314' 100300  030014                          STABLE  (SSSTRT,3,3,14)
004315' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

004316' 140000  011117'                 MA42:   MSG!TXNOT![ASCIZ /PUSH did not push PC+1 onto stack/]
004317' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```

                                  ; Test microcode

004320' 000000  050000            T42M:   MWORD   <ADDR=0,CJS,J=5>             ; push 1 onto stack     (0)
004321' 000000  000020
004322' 000100  070000                    MWORD   <CJS,J=7>                    ; push 2 onto stack     (1)
004323' 000000  000020
004324' 000200  110000                    MWORD   <CJS,J=11>                   ; push 3 onto stack     (2)
004325' 000000  000020
004326' 000300  130000                    MWORD   <CJS,J=13>                   ; push 4 onto stack     (3)
004327' 000000  000020
004330' 000400  000000                    MWORD   <JZ>                         ; initialize stack      (4)
004331' 000000  000000

004332' 000500  770000                    MWORD   <PUSH,DISA,CCOF,J=77>        ; disa/off      (5)
004333' 000000  010100
004334' 000600  000000                    MWORD   <CRTN>                       ; return to 5   (6)
004335' 000000  000240
004336' 000700  770000                    MWORD   <PUSH,DISA,CCON,J=77>        ; disa/on       (7)
004337' 000000  030100
004340' 001000  000000                    MWORD   <CRTN>                       ; return to 5   (10)
004341' 000000  000240
004342' 001100  770000                    MWORD   <PUSH,ENA,CCOF,J=77>         ; ena/off       (11)
004343' 000400  010100
004344' 001200  000000                    MWORD   <CRTN>                       ; return to 5   (12)
004345' 000000  000240
004346' 001300  770000                    MWORD   <PUSH,ENA,CCON,J=77>         ; ena/on        (13)
004347' 000400  030100
004350' 001400  000000                    MWORD   <CRTN>                       ; return to 5   (14)
004351' 000000  000240

004352' 007700  770000                    MWORD   <ADDR=77,JMAP,J=77>          ; error if here (77)
004353' 000000  000040
004354' 777777  777777                    -1
```

```

                                        ;#*****************************************************************
                                        ;* Test 43 - PUSH Instruction
                                        ;
                                        ; Description:  Verify  that  the   PUSH  instruction  loads  the
                                        ;               register/counter  with  the  data  on  the  direct
                                        ;               inputs  when TEST  inputs  are  satisfied.  Also,
                                        ;               verify that initial  register/counter contents do
                                        ;               not affect anything.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode
                                        ;               Set up RAR to start at location 100
                                        ;               Set MPRUN/SINGLE STEP (execute JZ)
                                        ;               Set MPRUN/SINGLE STEP (execute LDCT,PUSH,JRP)
                                        ;               Set MPRUN/SINGLE STEP (execute PUSH,JRP)
                                        ;               Set MPRUN/SINGLE STEP (execute PUSH,JRP)
                                        ;               Read LAR, and verify address = 12
                                        ;
                                        ; Failure:      ---
                                        ;#*****************************************************************

                                        ; Test data

 004355' 254 00 0 00 004366'            TSTS43: JRST    TG43                    ; go start test
 004356' 240401  000043                         SEQ!MPROC!NDMP!ZSEQ!43          ; test mask
 004357' 004414' 011113'                        T43M,,[ASCIZ ^PUSH Instruction^]
 004360' 010442' 010517'                        [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
 004361' 000000  004447'                        TSTS44                          ; failure test table
 004362' 000000  004527'                        TSTS45                          ;   ...
 004363' 000000  004616'                        TSTS46
 004364' 000000  004712'                        TSTS47
 004365' 777777  777777                         -1

                                        ; Start test

 004366' 201 00 0 00 000000'            TG43:   MOVEI   Z3                      ; get address of module start
 004367' 260 17 0 00 004270*                    GO      TRACE                   ; handle trace output
 004370' 201 01 0 00 004414'                    MOVEI   1,T43M                  ; set up microcode address
 004371' 260 17 0 00 004272*                    GO      TLOAD                   ; load/verify it
 004372' 263 17 0 00 000000                     RTN                             ; failed - exit test

                                        ; Initialization

 004373' 260 17 0 00 004274*            TL43:   GO      IPACLR                  ; clear port
 004374' 402 00 0 00 004275*                    SETZM   TSTSUB                  ; initialize subtest number
 004375' 201 06 0 00 004410'                    MOVEI   6,TS43                  ; get sstep table address
```

```

                                        ; Loop on single step table entries

      004376' 400 15 0 00 000000        TA43:   SETZ    ERFLG,                  ; clear error flag
      004377' 260 17 0 00 004300*               GO      SEXEC                   ; execute table entry
      004400' 254 00 0 00 004407'               JRST    TX43                    ; end of sstep table
      004401' 254 00 0 00 004376'               JRST    TA43                    ; keep looping after call
      004402' 474 15 0 00 000000                SETO    ERFLG,                  ; error - end address incorrect

                                        ; Handle error printouts and scope looping

      004403' 027 00 0 00 004412'               SCOPER  MA43                    ; print error message
      004404' 254 00 0 00 004373'               JRST    TL43                    ; loop on error
      004405' 254 00 0 00 004407'               JRST    TX43                    ; altmode exit
      004406' 322 15 0 00 004376'               JUMPE   ERFLG,TA43              ; do next sstep table entry

                                        ; End of test

      004407' 263 17 0 00 000000        TX43:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

      004410' 101001  000012            TS43:   STABLE  (SSSTRT,^D8,100,12)
      004411' 000000  000000                    STABLE  (SSLAST)

                                        ; Error messages

      004412' 140000  011126'           MA43:   MSG!TXNOT![ASCIZ /PUSH did not load cntr properly/]
      004413' 000000000000#                     LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```
                                ; Test microcode

004414' 010000  000000          T43M:   MWORD   <ADDR=100,JZ>               ; init PC               (100)
004415' 000000  000000
004416' 000000  000000                  MWORD   <ADDR=0,LDCT,J=0>           ; zero counter          (0)
004417' 000000  000300
004420' 000100  040000                  MWORD   <PUSH,DISA,CCOF,J=4>        ; load cntr with 4      (1)
004421' 000000  010100
004422' 000200  200000                  MWORD   <JRP,ENA,CCOF,J=20>         ; go to cntr addr       (2)
004423' 000400  010160
004424' 000300  030000                  MWORD   <JMAP,J=3>                  ; error if here         (3)
004425' 000000  000040
004426' 000400  070000                  MWORD   <PUSH,DISA,CCOF,J=7>        ; load cntr with 7      (4)
004427' 000000  010100
004430' 000500  200000                  MWORD   <JRP,ENA,CCOF,J=20>         ; go to cntr addr       (5)
004431' 000400  010160
004432' 000600  060000                  MWORD   <JMAP,J=6>                  ; error if here         (6)
004433' 000000  000040
004434' 000700  120000                  MWORD   <PUSH,DISA,CCOF,J=12>       ; load cntr with 12     (7)
004435' 000000  010100
004436' 001000  200000                  MWORD   <JRP,ENA,CCOF,J=20>         ; go to cntr addr       (10)
004437' 000400  010160
004440' 001100  110000                  MWORD   <JMAP,J=11>                 ; error if here         (11)
004441' 000000  000040
004442' 001200  000000                  MWORD   <CONT>                      ; done                  (12)
004443' 000000  000340

004444' 002000  200000                  MWORD   <ADDR=20,JMAP,J=20>         ; error if here         (20)
004445' 000000  000040
004446' 777777  777777                  -1
```

```
                                        ;#**************************************************************************
                                        ;* Test 44 - PUSH Instruction
                                        ;
                                        ; Description:  Verify that the PUSH instruction  does  not  load
                                        ;               the  register/counter  with  data  on  the direct
                                        ;               inputs  when  TEST  is  negated.  Also,  that the
                                        ;               initial  register/counter  contents do not affect
                                        ;               the instruction.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode
                                        ;               Set up RAR to start at location 100
                                        ;               Set MPRUN/SINGLE STEP (execute JZ)
                                        ;               Set MPRUN/SINGLE STEP (execute LDCT,PUSH,JRP)
                                        ;               Set MPRUN/SINGLE STEP (execute LDCT,PUSH,JRP)
                                        ;               Read LAR, and verify address = 0
                                        ;
                                        ; Failure:      ---
                                        ;#**************************************************************************

                                        ; Test data

004447' 254 00 0 00 004460'             TSTS44: JRST    TG44                    ; go start test
004450' 240401  000044                          SEQ!MPROC!NDMP!ZSEQ!44          ; test mask
004451' 004506' 011113'                         T44M,,[ASCIZ ^PUSH Instruction^]
004452' 010442' 010517'                         [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
004453' 000000  004527'                         TSTS45                          ; failure test table
004454' 000000  004616'                         TSTS46                          ;   ...
004455' 000000  004712'                         TSTS47
004456' 000000  005002'                         TSTS50
004457' 777777  777777                          -1

                                        ; Start test

004460' 201 00 0 00 000000'             TG44:   MOVEI   Z3                      ; get address of module start
004461' 260 17 0 00 004367*                     GO      TRACE                   ; handle trace output
004462' 201 01 0 00 004506'                     MOVEI   1,T44M                  ; set up microcode address
004463' 260 17 0 00 004371*                     GO      TLOAD                   ; load/verify it
004464' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

004465' 260 17 0 00 004373*             TL44:   GO      IPACLR                  ; clear port
004466' 402 00 0 00 004374*                     SETZM   TSTSUB                  ; initialize subtest number
004467' 201 06 0 00 004502'                     MOVEI   6,TS44                  ; get sstep table address
```

```

                                          ; Loop on single step table entries

004470' 400 15 0 00 000000      TA44:   SETZ    ERFLG,                  ; clear error flag
004471' 260 17 0 00 004377*             GO      SEXEC                   ; execute table entry
004472' 254 00 0 00 004501'             JRST    TX44                    ; end of sstep table
004473' 254 00 0 00 004470'             JRST    TA44                    ; keep looping after call
004474' 474 15 0 00 000000              SETO    ERFLG,                  ; error - end address incorrect

                                ; Handle error printouts and scope looping

004475' 027 00 0 00 004504'             SCOPER  MA44                    ; print error message
004476' 254 00 0 00 004465'             JRST    TL44                    ; loop on error
004477' 254 00 0 00 004501'             JRST    TX44                    ; altmode exit
004500' 322 15 0 00 004470'             JUMPE   ERFLG,TA44              ; do next sstep table entry

                                ; End of test

004501' 263 17 0 00 000000      TX44:   RTN                             ; return

                                ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

004502' 100701  000000          TS44:   STABLE  (SSSTRT,7,100,0)
004503' 000000  000000                  STABLE  (SSLAST)

                                ; Error messages

004504' 140000  011126'         MA44:   MSG!TXNOT![ASCIZ /PUSH did not load cntr properly/]
004505' 000000000000#                   LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                ; Test microcode

004506' 010000  000000          T44M:   MWORD   <ADDR=100,JZ>                   ; init PC               (100)
004507' 000000  000000
004510' 000000  040000                  MWORD   <ADDR=0,LDCT,J=4>               ; load counter          (0)
004511' 000000  000300
004512' 000100  030000                  MWORD   <PUSH,ENA,CCOF,J=3>             ; ena/off (don't load)  (1)
004513' 000400  010100
004514' 000200  030000                  MWORD   <JRP,ENA,CCOF,J=3>              ; go to cntr addr (4)   (2)
004515' 000400  010160
004516' 000300  030000                  MWORD   <JMAP,J=3>                      ; error if here         (3)
004517' 000000  000040

004520' 000400  000000                  MWORD   <LDCT,J=0>                      ; load counter          (4)
004521' 000000  000300
004522' 000500  030000                  MWORD   <PUSH,ENA,CCOF,J=3>             ; ena/off (don't load)  (5)
004523' 000400  010100
004524' 000600  030000                  MWORD   <JRP,ENA,CCOF,J=3>              ; go to cntr addr (0)   (6)
004525' 000400  010160
004526' 777777  777777                  -1
```

```

                                 ;#*******************************************************************
                                 ;* Test 45 - LOOP Instruction
                                 ;
                                 ; Description:  Verify that the LOOP instruction causes a jump to
                                 ;               the address specified at top  of stack if TEST is
                                 ;               negated.
                                 ;
                                 ; Procedure:    Clear Port
                                 ;               Load test microcode
                                 ;
                                 ;               Set up RAR to start at location 100
                                 ;               Set MPRUN/SINGLE STEP (execute JMAP,PUSH,PUSH,PUSH)
                                 ;               Set MPRUN/SINGLE STEP (execute LOOP,LOOP)
                                 ;               Read LAR, and verify address = 3
                                 ;
                                 ;               Set up RAR to start at location 101
                                 ;               Set MPRUN/SINGLE STEP (execute JMAP,PUSH,PUSH,PUSH)
                                 ;               Set MPRUN/SINGLE STEP (execute LOOP,LOOP)
                                 ;               Read LAR, and verify address = 7
                                 ;
                                 ; Failure:      ---
                                 ;#*******************************************************************

                                 ; Test data

004527' 254 00 0 00 004540'      TSTS45: JRST    TG45                    ; go start test
004530' 240401  000045                   SEQ!MPROC!NDMP!ZSEQ!45          ; test mask
004531' 004567' 011135'                  T45M,,[ASCIZ ^LOOP Instruction^]
004532' 010442' 010517'                  [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
004533' 000000  004616'                  TSTS46                          ; failure test table
004534' 000000  004712'                  TSTS47                          ;   ...
004535' 000000  005002'                  TSTS50
004536' 000000  005366'                  TSTS51
004537' 777777  777777                   -1

                                 ; Start test

004540' 201 00 0 00 000000'      TG45:   MOVEI   Z3                      ; get address of module start
004541' 260 17 0 00 004461*              GO      TRACE                   ; handle trace output
004542' 201 01 0 00 004567'              MOVEI   1,T45M                  ; set up microcode address
004543' 260 17 0 00 004463*              GO      TLOAD                   ; load/verify it
004544' 263 17 0 00 000000               RTN                             ; failed - exit test

                                 ; Initialization

004545' 260 17 0 00 004465*      TL45:   GO      IPACLR                  ; clear port
004546' 402 00 0 00 004466*              SETZM   TSTSUB                  ; initialize subtest number
004547' 201 06 0 00 004562'              MOVEI   6,TS45                  ; get sstep table address
```

```

                                        ; Loop on single step table entries

004550' 400 15 0 00 000000              TA45:   SETZ    ERFLG.                  ; clear error flag
004551' 260 17 0 00 004471*                     GO      SEXEC                   ; execute table entry
004552' 254 00 0 00 004561'                     JRST    TX45                    ; end of sstep table
004553' 254 00 0 00 004550'                     JRST    TA45                    ; keep looping after call
004554' 474 15 0 00 000000                      SETO    ERFLG.                  ; error - end address incorrect

                                        ; Handle error printouts and scope looping

004555' 027 00 0 00 004565'                     SCOPER  MA45                    ; print error message
004556' 254 00 0 00 004545'                     JRST    TL45                    ; loop on error
004557' 254 00 0 00 004561'                     JRST    TX45                    ; altmode exit
004560' 322 15 0 00 004550'                     JUMPE   ERFLG,TA45              ; do next sstep table entry

                                        ; End of test

004561' 263 17 0 00 000000              TX45:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

004562' 100601  000003                  TS45:   STABLE  (SSSTRT,6,100,3)
004563' 100601  010007                          STABLE  (SSSTRT,6,101,7)
004564' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

004565' 140000  011141'                 MA45:   MSG!TXNOT![ASCIZ /LOOP did not dispatch to stack address/]
004566' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```

                                       ; Test microcode

                                       ;       First do cntr = 0

004567' 010000  000000                 T45M:   MWORD   <ADDR=100,JMAP,J=0>          ; init PC               (100)
004570' 000000  000040
004571' 000000  000000                         MWORD   <ADDR=0,PUSH,J=0>            ; load cntr with 0      (0)
004572' 000000  000100
004573' 000100  100000                         MWORD   <PUSH,ENA,CCOF,J=10>         ; don't load cntr       (1)
004574' 000400  010100
004575' 000200  100000                         MWORD   <PUSH,ENA,CCOF,J=10>         ; don't load cntr       (2)
004576' 000400  010100
004577' 000300  440000                         MWORD   <LOOP,ENA,CCOF,J=44>         ; loop to stk contents  (3)
004600' 000400  010320

                                       ;       Then do cntr = 22

004601' 010100  040000                         MWORD   <ADDR=101,JMAP,J=4>          ; init PC               (101)
004602' 000000  000040
004603' 000400  220000                         MWORD   <ADDR=4,PUSH,J=22>           ; load cntr with 22     (4)
004604' 000000  000100
004605' 000500  100000                         MWORD   <PUSH,ENA,CCOF,J=10>         ; don't load cntr       (5)
004606' 000400  010100
004607' 000600  100000                         MWORD   <PUSH,ENA,CCOF,J=10>         ; don't load cntr       (6)
004610' 000400  010100
004611' 000700  440000                         MWORD   <LOOP,ENA,CCOF,J=44>         ; loop to stk contents  (7)
004612' 000400  010320

004613' 001000  100000                         MWORD   <JMAP,J=10>                  ; error if here         (10)
004614' 000000  000040
004615' 777777  777777                         -1
```

```
                                         ;#**********************************************************************
                                         ;* Test 46 - LOOP Instruction
                                         ;
                                         ; Description:   Verify that the LOOP instruction does a  continue
                                         ;                to next instruction if TEST is asserted, and that
                                         ;                register/counter  contents  are  neither affected
                                         ;                nor  effect  the  LOOP instruction.  Also, verify
                                         ;                that a LOOP instruction pops the stack correctly.
                                         ;
                                         ; Procedure:     Clear Port
                                         ;                Load test microcode
                                         ;                Set up RAR to start at location 100
                                         ;                Set MPRUN/SINGLE STEP (execute JZ,5 PUSH's,LDCT)
                                         ;                Set MPRUN/SINGLE STEP (execute LOOP,LOOP,LOOP)
                                         ;                Set MPRUN/SINGLE STEP (execute JRP,CRTN)
                                         ;                Read LAR, and verify address = 2
                                         ;
                                         ; Failure:       ---
                                         ;#**********************************************************************

                                         ; Test data

004616' 254 00 0 00 004627'              TSTS46: JRST    TG46                    ; go start test
004617' 240401  000046                           SEQ!MPROC!NDMP!ZSEQ!46          ; test mask
004620' 004655' 011135'                          T46M,,[ASCIZ ^LOOP Instruction^]
004621' 010442' 010517'                          [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
004622' 000000  004712'                          TSTS47                          ; failure test table
004623' 000000  005002'                          TSTS50                          ;   ...
004624' 000000  005366'                          TSTS51
004625' 000000  005456'                          TSTS52
004626' 777777  777777                           -1

                                         ; Start test

004627' 201 00 0 00 000000'              TG46:   MOVEI   Z3                      ; get address of module start
004630' 260 17 0 00 004541*                      GO      TRACE                   ; handle trace output
004631' 201 01 0 00 004655'                      MOVEI   1,T46M                  ; set up microcode address
004632' 260 17 0 00 004543*                      GO      TLOAD                   ; load/verify it
004633' 263 17 0 00 000000                       RTN                             ; failed - exit test

                                         ; Initialization

004634' 260 17 0 00 004545*              TL46:   GO      IPACLR                  ; clear port
004635' 402 00 0 00 004546*                      SETZM   TSTSUB                  ; initialize subtest number
004636' 201 06 0 00 004651'                      MOVEI   6,TS46                  ; get sstep table address
```

```

                                        ; Loop on single step table entries

004637' 400 15 0 00 000000      TA46:   SETZ    ERFLG,                  ; clear error flag
004640' 260 17 0 00 004551*             GO      SEXEC                   ; execute table entry
004641' 254 00 0 00 004650'             JRST    TX46                    ; end of sstep table
004642' 254 00 0 00 004637'             JRST    TA46                    ; keep looping after call
004643' 474 15 0 00 000000              SETO    ERFLG,                  ; error - end address incorrect

                                        ; Handle error prirtouts and scope looping

004644' 027 00 0 00 004653'             SCOPER  MA46                    ; print error message
004645' 254 00 0 00 004634'             JRST    TL46                    ; loop on error
004646' 254 00 0 00 004650'             JRST    TX46                    ; altmode exit
004647' 322 15 0 00 004637'             JUMPE   ERFLG,TA46              ; do next sstep table entry

                                        ; End of test

004650' 263 17 0 00 000000      TX46:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

004651' 101401  000002          TS46:   STABLE  (SSSTRT,^D12,100,2)
004652' 000000  000000                  STABLE  (SSLAST)

                                        ; Error messages

004653' 140000  011151'         MA46:   MSG!TXNOT![ASCIZ /LOOP broken/]
004654' 000000000000#                   LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```
                                        ; Test microcode

004655' 000000  000000          T46M:   MWORD   <ADDR=0,PUSH>                   ; push 1 onto stack     (0)
004656' 000000  000100
004657' 000100  000000                  MWORD   <PUSH>                          ; push 2 onto stack     (1)
004660' 000000  000100
004661' 000200  000000                  MWORD   <PUSH>                          ; push 3 onto stack     (2)
004662' 000000  000100
004663' 000300  000000                  MWORD   <PUSH>                          ; push 4 onto stack     (3)
004664' 000000  000100
004665' 000400  000000                  MWORD   <PUSH>                          ; push 5 onto stack     (4)
004666' 000000  000100
004667' 000501  010C00                  MWORD   <LDCT,J=101>                    ; load counter          (5)
004670' 000000  000300
004671' 000600  770000                  MWORD   <LOOP,DISA,CCOF,J=77>           ; disa/off              (6)
004672' 000000  010320
004673' 000700  770000                  MWORD   <LOOP,DISA,CCON,J=77>           ; disa/on               (7)
004674' 000000  030320
004675' 001000  770000                  MWORD   <LOOP,ENA,CCON,J=77>            ; ena/on                (10)
004676' 000400  030320
004677' 001100  120000                  MWORD   <JRP,ENA,CCOF,J=12>             ; go to cntr addr (101) (11)
004700' 000400  010160
004701' 001200  150000                  MWORD   <JMAP,J=15>                     ; error if here         (12)
004702' 000000  000040

004703' 007700  770000                  MWORD   <ADDR=77,JMAP,J=77>             ; error if here         (77)
004704' 000000  000040
004705' 010000  000000                  MWORD   <JZ>                            ; initialize stack      (100)
004706' 000000  000000
004707' 010100  000000                  MWORD   <CRTN>                          ; return to             (101)
004710' 000000  000240
004711' 777777  777777                  -1
```

```
                                    ;#*******************************************************************
                                    ;* Test 47 - LOOP Instruction
                                    ;
                                    ; Description:  Verify that the LOOP instruction does not pop the
                                    ;               stack if TEST is negated.
                                    ;
                                    ; Procedure:    Clear Port
                                    ;               Load test microcode
                                    ;               Set up RAR to start at location 100
                                    ;               Set MPRUN/SINGLE STEP (execute JZ,4 PUSH's)
                                    ;               Set MPRUN/SINGLE STEP (execute LDCT,PUSH)
                                    ;               Set MPRUN/SINGLE STEP (execute RPCT,LOOP 3 times)
                                    ;               Set MPRUN/SINGLE STEP (execute RPCT,JMAP,CRTN)
                                    ;               Read LAR, and verify address = 6
                                    ;
                                    ; Failure:      ---
                                    ;#*******************************************************************

                                    ; Test data

004712' 254 00 0 00 004723'         TSTS47: JRST    TG47                    ; go start test
004713' 240401  000047                      SEQ!MPROC!NDMP!ZSEQ!47          ; test mask
004714' 004751' 011135'                     T47M,,[ASCIZ ^LOOP Instruction^]
004715' 010442' 010517'                     [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
004716' 000000  005002'                     TSTS50                          ; failure test table
004717' 000000  005366'                     TSTS51                          ;   ...
004720' 000000  005456'                     TSTS52
004721' 000000  005653'                     TSTS53
004722' 777777  777777                      -1

                                    ; Start test

004723' 201 00 0 00 000000'         TG47:   MOVEI   Z3                      ; get address of module start
004724' 260 17 0 00 004630*                 GO      TRACE                   ; handle trace output
004725' 201 01 0 00 004751'                 MOVEI   1,T47M                  ; set up microcode address
004726' 260 17 0 00 004632*                 GO      TLOAD                   ; load/verify it
004727' 263 17 0 00 000000                  RTN                             ; failed - exit test

                                    ; Initialization

004730' 260 17 0 00 004634*         TL47:   GO      IPACLR                  ; clear port
004731' 402 00 0 00 004635*                 SETZM   TSTSUB                  ; initialize subtest number
004732' 201 06 0 00 004745'                 MOVEI   6,TS47                  ; get sstep table address

                                    ; Loop on single step table entries

004733' 400 15 0 00 000000          TA47:   SETZ    ERFLG,                  ; clear error flag
004734' 260 17 0 00 004640*                 GO      SEXEC                   ; execute table entry
004735' 254 00 0 00 004744'                 JRST    TX47                    ; end of sstep table
004736' 254 00 0 00 004733'                 JRST    TA47                    ; keep looping after call
004737' 474 15 0 00 000000                  SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

004740' 027 00 0 00 004747'                     SCOPER  MA47                    ; print error message
004741' 254 00 0 00 004730'                     JRST    TL47                    ; loop on error
004742' 254 00 0 00 004744'                     JRST    TX47                    ; altmode exit
004743' 322 15 0 00 004733'                     JUMPE   ERFLG,TA47              ; do next sstep table entry

                                        ; End of test

004744' 263 17 0 00 000000              TX47:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

004745' 102001  000006                  TS47:   STABLE  (SSSTRT,^D16,100,6)
004746' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

004747' 140000  011154'                 MA47:   MSG!TXNOT![ASCIZ /LOOP affected the stack when it shouldn't have/]
004750' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

004751' 000000  000000                  T47M:   MWORD   <ADDR=0,PUSH>                   ; push 1 onto stack     (0)
004752' 000000  000100
004753' 000100  000000                          MWORD   <PUSH>                          ; push 2 onto stack     (1)
004754' 000000  000100
004755' 000200  000000                          MWORD   <PUSH>                          ; push 3 onto stack     (2)
004756' 000000  000100
004757' 000300  000000                          MWORD   <PUSH>                          ; push 4 onto stack     (3)
004760' 000000  000100
004761' 000400  030000                          MWORD   <LDCT,J=3>                      ; load loop count       (4)
004762' 000000  000300
004763' 000500  000000                          MWORD   <PUSH,ENA,CCOF>                 ; push 6 onto stack     (5)
004764' 000400  010100
004765' 000600  100000                          MWORD   <RPCT,J=10>                     ; count down, go to 7   (6)
004766' 000000  000220
004767' 000700  200000                          MWORD   <JMAP,J=20>                     ; get out of loop       (7)
004770' 000000  000040
004771' 001000  440000                          MWORD   <LOOP,ENA,CCOF,J=44>            ; ena/off               (10)
004772' 000400  010320
004773' 001100  110000                          MWORD   <JMAP,J=11>                     ; error if here         (11)
004774' 000000  000040

004775' 002000  000000                          MWORD   <ADDR=20,CRTN>                  ; return to 6           (20)
004776' 000000  000240

004777' 010000  000000                          MWORD   <ADDR=100,JZ>                   ; initialize stack      (100)
005000' 000000  000000
005001' 777777  777777                          -1
```

```
                                        ;#*********************************************************************
                                        ;* Test 50 - RFCT Instruction
                                        ;
                                        ; Description:  Verify that the RFCT instruction causes a jump to
                                        ;               the address  specified  at top  of  stack  if the
                                        ;               counter is not = 0.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode
                                        ;               Set up RAR to start at location 333
                                        ;               Set MPRUN/SINGLE STEP (Execute JZ)
                                        ;               Repeat for each data pattern (12 of them,
                                        ;                 floating 1's):  Execute 4(LDCT,CJS,RFCT)
                                        ;               Read LAR, and verify address = 74
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

 005002' 254 00 0 00 005013'            TSTS50: JRST    TG50                    ; go start test
 005003' 240401  000050                         SEQ!MPROC!NDMP!ZSEQ!50          ; test mask
 005004' 005041' 011166'                        T50M,,[ASCIZ ^RFCT Instruction^]
 005005' 010442' 010517'                        [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
 005006' 000000  005366'                        TSTS51                          ; failure test table
 005007' 000000  005456'                        TSTS52                          ;   ...
 005010' 000000  005653'                        TSTS53
 005011' 000000  006275'                        TSTS54
 005012' 777777  777777                         -1

                                        ; Start test

 005013' 201 00 0 00 000000'            TG50:   MOVEI   Z3                      ; get address of module start
 005014' 260 17 0 00 004724*                    GO      TRACE                   ; handle trace output
 005015' 201 01 0 00 005041'                    MOVEI   1,T50M                  ; set up microcode address
 005016' 260 17 0 00 004726*                    GO      TLOAD                   ; load/verify it
 005017' 263 17 0 00 000000                     RTN                             ; failed - exit test

                                        ; Initialization

 005020' 260 17 0 00 004730*            TL50:   GO      IPACLR                  ; clear port
 005021' 402 00 0 00 004731*                    SETZM   TSTSUB                  ; initialize subtest number
 005022' 201 06 0 00 005035'                    MOVEI   6,TS50                  ; get sstep table address

                                        ; Loop on single step table entries

 005023' 400 15 0 00 000000             TA50:   SETZ    ERFLG,                  ; clear error flag
 005024' 260 17 0 00 004734*                    GO      SEXEC                   ; execute table entry
 005025' 254 00 0 00 005034'                    JRST    TX50                    ; end of sstep table
 005026' 254 00 0 00 005023'                    JRST    TA50                    ; keep looping after call
 005027' 474 15 0 00 000000                     SETO    ERFLG,                  ; error - end address incorrect
```

```


                                  ; Handle error printouts and scope looping

005030' 027 00 0 00 005037'               SCOPER  MA50                    ; print error message
005031' 254 00 0 00 005020'               JRST    TL50                    ; loop on error
005032' 254 00 0 00 005034'               JRST    TX50                    ; altmode exit
005033' 322 15 0 00 005023'               JUMPE   ERFLG,TA50              ; do next sstep table entry

                                  ; End of test

005034' 263 17 0 00 000000        TX50:   RTN                             ; return

                                  ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

005035' 122103  330140            TS50:   STABLE  (SSSTRT,^D145,333,140)
005036' 000000  000000                    STABLE  (SSLAST)

                                  ; Error messages

005037' 140000  011172'           MA50:   MSG!TXNOT![ASCIZ /RFCT did not dispatch to stack address/]
005040' 000000000000#                     LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                  ; Test microcode

005041' 033300  000000            T50M:   MWORD   <ADDR=333,JZ>                   ; init PC       (333)
005042' 000000  000000

005043' 000000  010000                    MWORD   <ADDR=0,LDCT,J=1>               ; load counter  (0)
005044' 000000  000300
005045' 000110  000000                    MWORD   <CJS,J=1000>                    ; push/gosub    (1)
005046' 000000  000020
005047' 000200  010000                    MWORD   <LDCT,J=1>                      ; load counter  (2)
005050' 000000  000300
005051' 000310  020000                    MWORD   <CJS,J=1002>                    ; push/gosub    (3)
005052' 000000  000020
005053' 000400  010000                    MWORD   <LDCT,J=1>                      ; load counter  (4)
005054' 000000  000300
005055' 000510  040000                    MWORD   <CJS,J=1004>                    ; push/gosub    (5)
005056' 000000  000020
005057' 000600  010000                    MWORD   <LDCT,J=1>                      ; load counter  (6)
005060' 000000  000300
005061' 000710  060000                    MWORD   <CJS,J=1006>                    ; push/gosub    (7)
005062' 000000  000020

005063' 001000  020000                    MWORD   <LDCT,J=2>                      ; load counter  (10)
005064' 000000  000300
005065' 001110  000000                    MWORD   <CJS,J=1000>                    ; push/gosub    (11)
005066' 000000  000020
005067' 001200  020000                    MWORD   <LDCT,J=2>                      ; load counter  (12)
005070' 000000  000300
005071' 001310  020000                    MWORD   <CJS,J=1002>                    ; push/gosub    (13)
005072' 000000  000020
005073' 001400  020000                    MWORD   <LDCT,J=2>                      ; load counter  (14)
005074' 000000  000300
005075' 001510  040000                    MWORD   <CJS,J=1004>                    ; push/gosub    (15)
```

```
 4583   005076' 000000  000020
 4584   005077' 001600  020000                  MWORD   <LDCT,J=2>                      ; load counter  (16)
 4585   005100' 000000  000300
 4586   005101' 001710  060000                  MWORD   <CJS,J=1006>                    ; push/gosub    (17)
 4587   005102' 000000  000020
 4588
 4589   005103' 002000  040000                  MWORD   <LDCT,J=4>                      ; load counter  (20)
 4590   005104' 000000  000300
 4591   005105' 002110  000000                  MWORD   <CJS,J=1000>                    ; push/gosub    (21)
 4592   005106' 000000  000020
 4593   005107' 002200  040000                  MWORD   <LDCT,J=4>                      ; load counter  (22)
 4594   005110' 000000  000300
 4595   005111' 002310  020000                  MWORD   <CJS,J=1002>                    ; push/gosub    (23)
 4596   005112' 000000  000020
 4597   005113' 002400  040000                  MWORD   <LDCT,J=4>                      ; load counter  (24)
 4598   005114' 000000  000300
 4599   005115' 002510  040000                  MWORD   <CJS,J=1004>                    ; push/gosub    (25)
 4600   005116' 000000  000020
 4601   005117' 002600  040000                  MWORD   <LDCT,J=4>                      ; load counter  (26)
 4602   005120' 000000  000300
 4603   005121' 002710  060000                  MWORD   <CJS,J=1006>                    ; push/gosub    (27)
 4604   005122' 000000  000020
 4605
 4606   005123' 003000  100000                  MWORD   <LDCT,J=10>                     ; load counter  (30)
 4607   005124' 000000  000300
 4608   005125' 003110  000000                  MWORD   <CJS,J=1000>                    ; push/gosub    (31)
 4609   005126' 000000  000020
 4610   005127' 003200  100000                  MWORD   <LDCT,J=10>                     ; load counter  (32)
 4611   005130' 000000  000300
 4612   005131' 003310  020000                  MWORD   <CJS,J=1002>                    ; push/gosub    (33)
 4613   005132' 000000  000020
 4614   005133' 003400  100000                  MWORD   <LDCT,J=10>                     ; load counter  (34)
 4615   005134' 000000  000300
 4616   005135' 003510  040000                  MWORD   <CJS,J=1004>                    ; push/gosub    (35)
 4617   005136' 000000  000020
 4618   005137' 003600  100000                  MWORD   <LDCT,J=10>                     ; load counter  (36)
 4619   005140' 000000  000300
 4620   005141' 003710  060000                  MWORD   <CJS,J=1006>                    ; push/gosub    (37)
 4621   005142' 000000  000020
 4622
 4623   005143' 004000  200000                  MWORD   <LDCT,J=20>                     ; load counter  (40)
 4624   005144' 000000  000300
 4625   005145' 004110  000000                  MWORD   <CJS,J=1000>                    ; push/gosub    (41)
 4626   005146' 000000  000020
 4627   005147' 004200  200000                  MWORD   <LDCT,J=20>                     ; load counter  (42)
 4628   005150' 000000  000300
 4629   005151' 004310  020000                  MWORD   <CJS,J=1002>                    ; push/gosub    (43)
 4630   005152' 000000  000020
 4631   005153' 004400  200000                  MWORD   <LDCT,J=20>                     ; load counter  (44)
 4632   005154' 000000  000300
 4633   005155' 004510  040000                  MWORD   <CJS,J=1004>                    ; push/gosub    (45)
 4634   005156' 000000  000020
 4635   005157' 004600  200000                  MWORD   <LDCT,J=20>                     ; load counter  (46)
 4636   005160' 000000  000300
 4637   005161' 004710  060000                  MWORD   <CJS,J=1006>                    ; push/gosub    (47)
```

```
005162' 000000  000020

005163' 005000  400000          MWORD   <LDCT,J=40>             ; load counter  (50)
005164' 000000  000300
005165' 005110  000000          MWORD   <CJS,J=1000>            ; push/gosub    (51)
005166' 000000  000020
005167' 005200  400000          MWORD   <LDCT,J=40>             ; load counter  (52)
005170' 000000  000300
005171' 005310  020000          MWORD   <CJS,J=1002>            ; push/gosub    (53)
005172' 000000  000020
005173' 005400  400000          MWORD   <LDCT,J=40>             ; load counter  (54)
005174' 000000  000300
005175' 005510  040000          MWORD   <CJS,J=1004>            ; push/gosub    (55)
005176' 000000  000020
005177' 005600  400000          MWORD   <LDCT,J=40>             ; load counter  (56)
005200' 000000  000300
005201' 005710  060000          MWORD   <CJS,J=1006>            ; push/gosub    (57)
005202' 000000  000020

005203' 006001  000000          MWORD   <LDCT,J=100>            ; load counter  (60)
005204' 000000  000300
005205' 006110  000000          MWORD   <CJS,J=1000>            ; push/gosub    (61)
005206' 000000  000020
005207' 006201  000000          MWORD   <LDCT.J=100>            ; load counter  (62)
005210' 000000  000300
005211' 006310  020000          MWORD   <CJS,J=1002>            ; push/gosub    (63)
005212' 000000  000020
005213' 006401  000000          MWORD   <LDCT,J=100>            ; load counter  (64)
005214' 000000  000300
005215' 006510  040000          MWORD   <CJS,J=1004>            ; push/gosub    (65)
005216' 000000  000020
005217' 006601  000000          MWORD   <LDCT,J=100>            ; load counter  (66)
005220' 000000  000300
005221' 006710  060000          MWORD   <CJS,J=1006>            ; push/gosub    (67)
005222' 000000  000020

005223' 007002  000000          MWORD   <LDCT,J=200>            ; load counter  (70)
005224' 000000  000300
005225' 007110  000000          MWORD   <CJS,J=1000>            ; push/gosub    (71)
005226' 000000  000020
005227' 007202  000000          MWORD   <LDCT,J=200>            ; load counter  (72)
005230' 000000  000300
005231' 007310  020000          MWORD   <CJS,J=1002>            ; push/gosub    (73)
005232' 000000  000020
005233' 007402  000000          MWORD   <LDCT,J=200>            ; load counter  (74)
005234' 000000  000300
005235' 007510  040000          MWORD   <CJS,J=1004>            ; push/gosub    (75)
005236' 000000  000020
005237' 007602  000000          MWORD   <LDCT,J=200>            ; load counter  (76)
005240' 000000  000300
005241' 007710  060000          MWORD   <CJS,J=1006>            ; push/gosub    (77)
005242' 000000  000020

005243' 010004  000000          MWORD   <LDCT,J=400>            ; load counter  (100)
005244' 000000  000300
```

```
4693  005245' 010110  000000          MWORD   <CJS,J=1000>                ; push/gosub     (101)
4694  005246' 000000  000020
4695  005247' 010204  000000          MWORD   <LDCT,J=400>                ; load counter   (102)
4696  005250' 000000  000300
4697  005251' 010310  020000          MWORD   <CJS,J=1002>                ; push/gosub     (103)
4698  005252' 000000  000020
4699  005253' 010404  000000          MWORD   <LDCT,J=400>                ; load counter   (104)
4700  005254' 000000  000300
4701  005255' 010510  040000          MWORD   <CJS,J=1004>                ; push/gosub     (105)
4702  005256' 000000  000020
4703  005257' 010604  000000          MWORD   <LDCT,J=400>                ; load counter   (106)
4704  005260' 000000  000300
4705  005261' 010710  060000          MWORD   <CJS,J=1006>                ; push/gosub     (107)
4706  005262' 000000  000020
4707
4708  005263' 011010  000000          MWORD   <LDCT,J=1000>               ; load counter   (110)
4709  005264' 000000  000300
4710  005265' 011110  000000          MWORD   <CJS,J=1000>                ; push/gosub     (111)
4711  005266' 000000  000020
4712  005267' 011210  000000          MWORD   <LDCT,J=1000>               ; load counter   (112)
4713  005270' 000000  000300
4714  005271' 011310  020000          MWORD   <CJS,J=1002>                ; push/gosub     (113)
4715  005272' 000000  000020
4716  005273' 011410  000000          MWORD   <LDCT,J=1000>               ; load counter   (114)
4717  005274' 000000  000300
4718  005275' 011510  040000          MWORD   <CJS,J=1004>                ; push/gosub     (115)
4719  005276' 000000  000020
4720  005277' 011610  000000          MWORD   <LDCT,J=1000>               ; load counter   (116)
4721  005300' 000000  000300
4722  005301' 011710  060000          MWORD   <CJS,J=1006>                ; push/gosub     (117)
4723  005302' 000000  000020
4724
4725  005303' 012020  000000          MWORD   <LDCT,J=2000>               ; load counter   (120)
4726  005304' 000000  000300
4727  005305' 012110  000000          MWORD   <CJS,J=1000>                ; push/gosub     (121)
4728  005306' 000000  000020
4729  005307' 012220  000000          MWORD   <LDCT,J=2000>               ; load counter   (122)
4730  005310' 000000  000300
4731  005311' 012310  020000          MWORD   <CJS,J=1002>                ; push/gosub     (123)
4732  005312' 000000  000020
4733  005313' 012420  000000          MWORD   <LDCT,J=2000>               ; load counter   (124)
4734  005314' 000000  000300
4735  005315' 012510  040000          MWORD   <CJS,J=1004>                ; push/gosub     (125)
4736  005316' 000000  000020
4737  005317' 012620  000000          MWORD   <LDCT,J=2000>               ; load counter   (126)
4738  005320' 000000  000300
4739  005321' 012710  060000          MWORD   <CJS,J=1006>                ; push/gosub     (127)
4740  005322' 000000  000020
4741
4742  005323' 013040  000000          MWORD   <LDCT,J=4000>               ; load counter   (130)
4743  005324' 000000  000300
4744  005325' 013110  000000          MWORD   <CJS,J=1000>                ; push/gosub     (131)
4745  005326' 000000  000020
4746  005327' 013240  000000          MWORD   <LDCT,J=4000>               ; load counter   (132)
4747  005330' 000000  000300
```

```
 4748  005331' 013310  020000              MWORD   <CJS,J=1002>                      ; push/gosub     (133)
 4749  005332' 000000  000020
 4750  005333' 013440  000000              MWORD   <LDCT,J=4000>                     ; load counter   (134)
 4751  005334' 000000  000300
 4752  005335' 013510  040000              MWORD   <CJS,J=1004>                      ; push/gosub     (135)
 4753  005336' 000000  000020
 4754  005337' 013640  000000              MWORD   <LDCT,J=4000>                     ; load counter   (136)
 4755  005340' 000000  000300
 4756  005341' 013710  060000              MWORD   <CJS,J=1006>                      ; push/gosub     (137)
 4757  005342' 000000  000020
 4758
 4759  005343' 077707  770000              MWORD   <ADDR=777,JMAP,J=777>             ; error if here          (777)
 4760  005344' 000000  000040
 4761  005345' 100007  770000              MWORD   <ADDR=1000,RFCT,DISA,CCOF,J=777>; return to stk addr     (1000)
 4762  005346' 000000  010200
 4763  005347' 100110  010000              MWORD   <JMAP,J=1001>                     ; error if here          (1001)
 4764  005350' 000000  000040
 4765  005351' 100207  770000              MWORD   <RFCT,DISA,CCON,J=777>            ; return to stack addr   (1002)
 4766  005352' 000000  030200
 4767  005353' 100310  030000              MWORD   <JMAP,J=1003>                     ; error if here          (1003)
 4768  005354' 000000  000040
 4769  005355' 100407  770000              MWORD   <RFCT,ENA,CCOF,J=777>             ; return to stack addr   (1004)
 4770  005356' 000400  010200
 4771  005357' 100510  050000              MWORD   <JMAP,J=1005>                     ; error if here          (1005)
 4772  005360' 000000  000040
 4773  005361' 100607  770000              MWORD   <RFCT,ENA,CCON,J=777>             ; return to stack addr   (1006)
 4774  005362' 000400  030200
 4775  005363' 100710  070000              MWORD   <JMAP,J=1007>                     ; error if here          (1007)
 4776  005364' 000000  000040
 4777  005365' 777777  777777              -1
```

```
                                  ;#********************************************************************
                                  ;* Test 51 - RFCT Instruction
                                  ;
                                  ; Description:  Verify that the RFCT instruction continues to the
                                  ;               next instruction if the counter = 0. Also, verify
                                  ;               that the stack is popped with each RFCT.
                                  ;
                                  ; Procedure:    Clear Port
                                  ;               Load test microcode
                                  ;               Set up RAR to start at location 100
                                  ;               Set MPRUN/SINGLE STEP (execute JZ,5 PUSH's)
                                  ;               Set MPRUN/SINGLE STEP (execute 4 RFCT's,CRTN)
                                  ;               Read LAR, and verify address = 1
                                  ;
                                  ; Failure:      ---
                                  ;#********************************************************************

                                  ; Test data

005366' 254 00 0 00 005377'       TSTS51: JRST    TG51                    ; go start test
005367' 240401  000051                    SEQ!MPROC!NDMP!ZSEQ!51          ; test mask
005370' 005425' 011166'                   T51M,,[ASCIZ ^RFCT Instruction^]
005371' 010442' 010517'                   [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
005372' 000000  005456'                   TSTS52                          ; failure test table
005373' 000000  005653'                   TSTS53                          ;   ...
005374' 000000  006275'                   TSTS54
005375' 000000  006377'                   TSTS55
005376' 777777  777777                    -1

                                  ; Start test

005377' 201 00 0 00 000000'       TG51:   MOVEI   Z3                      ; get address of module start
005400' 260 17 0 00 005014*               GO      TRACE                   ; handle trace output
005401' 201 01 0 00 005425'               MOVEI   1,T51M                  ; set up microcode address
005402' 260 17 0 00 005016*               GO      TLOAD                   ; load/verify it
005403' 263 17 0 00 000000                RTN                             ; failed - exit test

                                  ; Initialization

005404' 260 17 0 00 005020*       TL51:   GO      IPACLR                  ; clear port
005405' 402 00 0 00 005021*               SETZM   TSTSUB                  ; initialize subtest number
005406' 201 06 0 00 005421'               MOVEI   6,TS51                  ; get sstep table address

                                  ; Loop on single step table entries

005407' 400 15 0 00 000000        TA51:   SETZ    ERFLG,                  ; clear error flag
005410' 260 17 0 00 005024*               GO      SEXEC                   ; execute table entry
005411' 254 00 0 00 005420'               JRST    TX51                    ; end of sstep table
005412' 254 00 0 00 005407'               JRST    TA51                    ; keep looping after call
005413' 474 15 0 00 000000                SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

005414' 027 00 0 00 005423'                     SCOPER  MA51                    ; print error message
005415' 254 00 0 00 005404'                     JRST    TL51                    ; loop on error
005416' 254 00 0 00 005420'                     JRST    TX51                    ; altmode exit
005417' 322 15 0 00 005407'                     JUMPE   ERFLG,TA51              ; do next sstep table entry

                                        ; End of test

005420' 263 17 0 00 000000              TX51:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

005421' 101301  000001                  TS51:   STABLE  (SSSTRT,^D11,100,1)
005422' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

005423' 140000  011202'                 MA51:   MSG!TXNOT![ASCIZ /RFCT not continuing at PC+1 or not popping stack/]
005424' 00C000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

005425' 000000  000000                  T51M:   MWORD   <ADDR=0,PUSH>                   ; push 1 onto stack     (0)
005426' 000000  000100
005427' 000100  000000                          MWORD   <PUSH>                          ; push 2 onto stack     (1)
005430' 000000  000100
005431' 000200  000000                          MWORD   <PUSH>                          ; push 3 onto stack     (2)
005432' 000000  000100
005433' 000300  000000                          MWORD   <PUSH>                          ; push 4 onto stack     (3)
005434' 000000  000100
005435' 000400  000000                          MWORD   <PUSH,J=0>                      ; push 5/zero cntr      (4)
005436' 000000  000100
005437' 000500  770000                          MWORD   <RFCT,DISA,CCOF,J=77>           ; continue to 2         (5)
005440' 000000  010200
005441' 000600  770000                          MWORD   <RFCT,DISA,CCON,J=77>           ; continue to 3         (6)
005442' 000000  030200
005443' 000700  770000                          MWORD   <RFCT,ENA,CCOF,J=77>            ; continue to 4         (7)
005444' 000400  010200
005445' 001000  770000                          MWORD   <RFCT,ENA,CCON,J=77>            ; continue to 5         (10)
005446' 000400  030200
005447' 001100  000000                          MWORD   <CRTN>                          ; return to 1           (11)
005450' 000000  000240
005451' 007700  770000                          MWORD   <ADDR=77,JMAP,J=77>             ;                       (77)
005452' 000000  000040
005453' 010000  000000                          MWORD   <JZ>                            ; initialize stack      (100)
005454' 000000  000000
005455' 777777  777777                          -1
```

```

                                        ;#*********************************************************************
                                        ;* Test 52 - RFCT Instruction
                                        ;
                                        ; Description:  Verify that the RFCT instruction does not pop the
                                        ;               stack if the counter is nonzero.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode
                                        ;               Set up RAR to start at location 0
                                        ;               Set MPRUN/SINGLE STEP (execute JMAP,PUSH,PUSH)
                                        ;               Set MPRUN/SINGLE STEP (execute CJS,RFCT,CRTN)
                                        ;               Read LAR, and verify address = 4
                                        ;
                                        ;               Set up RAR to start at location 5
                                        ;               Set MPRUN/SINGLE STEP (execute JMAP,PUSH,PUSH)
                                        ;               Set MPRUN/SINGLE STEP (execute CJS,RFCT,CRTN)
                                        ;               Read LAR, and verify address = 11
                                        ;
                                        ;               Set up RAR to start at location 12
                                        ;               Set MPRUN/SINGLE STEP (execute JMAP,PUSH,PUSH)
                                        ;               Set MPRUN/SINGLE STEP (execute CJS,RFCT,CRTN)
                                        ;               Read LAR, and verify address = 16
                                        ;
                                        ;               Set up RAR to start at location 17
                                        ;               Set MPRUN/SINGLE STEP (execute JMAP,PUSH,PUSH)
                                        ;               Set MPRUN/SINGLE STEP (execute CJS,RFCT,CRTN)
                                        ;               Read LAR, and verify address = 23
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

005456' 254 00 0 00 005467'             TSTS52: JRST    TG52                    ; go start test
005457' 240401  000052                          SEQ!MPROC!NDMP!ZSEQ!52          ; test mask
005460' 005520' 011166'                         T52M,,[ASCIZ ^RFCT Instruction^]
005461' 010442' 010517'                         [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
005462' 000000  005653'                         TSTS53                          ; failure test table
005463' 000000  006275'                         TSTS54                          ;   ...
005464' 000000  006377'                         TSTS55
005465' 000000  006707'                         TSTS56
005466' 777777  777777                          -1

                                        ; Start test

005467' 201 00 0 00 000000'             TG52:   MOVEI   Z3                      ; get address of module start
005470' 260 17 0 00 005400*                     GO      TRACE                   ; handle trace output
005471' 201 01 0 00 005520'                     MOVEI   1,T52M                  ; set up microcode address
005472' 260 17 0 00 005402*                     GO      TLOAD                   ; load/verify it
005473' 263 17 0 00 000000                      RTN                             ; failed - exit test
```

```

                                         ; Initialization

005474' 260 17 0 00 005404*      TL52:   GO      IPACLR                  ; clear port
005475' 402 00 0 00 005405*              SETZM   TSTSUB                  ; initialize subtest number
005476' 201 06 0 00 005511'              MOVEI   6,TS52                  ; get sstep table address

                                         ; Loop on single step table entries

005477' 400 15 0 00 000000       TA52:   SETZ    ERFLG,                  ; clear error flag
005500' 260 17 0 00 005410*              GO      SEXEC                   ; execute table entry
005501' 254 00 0 00 005510'              JRST    TX52                    ; end of sstep table
005502' 254 00 0 00 005477'              JRST    TA52                    ; keep looping after call
005503' 474 15 0 00 000000               SETO    ERFLG,                  ; error - end address incorrect

                                         ; Handle error printouts and scope looping

005504' 027 00 0 00 005516'              SCOPER  MA52                    ; print error message
005505' 254 00 0 00 005474'              JRST    TL52                    ; loop on error
005506' 254 00 0 00 005510'              JRST    TX52                    ; altmode exit
005507' 322 15 0 00 005477'              JUMPE   ERFLG,TA52              ; do next sstep table entry

                                         ; End of test

005510' 263 17 0 00 000000       TX52:   RTN                             ; return

                                         ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

005511' 101300  000007           TS52:   STABLE  (SSSTRT,^D11,0,7)
005512' 101300  200027                   STABLE  (SSSTRT,^D11,20,27)
005513' 101300  400047                   STABLE  (SSSTRT,^D11,40,47)
005514' 101300  600067                   STABLE  (SSSTRT,^D11,60,67)
005515' 000000  000000                   STABLE  (SSLAST)

                                         ; Error messages

005516' 140000  011214'          MA52:   MSG!TXNOT![ASCIZ /RFCT affected stack with cntr nonzero/]
005517' 000000000000#                    LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```


                                  ; Test microcode
005520' 000000  010000          T52M:   MWORD   <ADDR=0,CJPP,J=1>          ; pop stack     (0)
005521' 000000  000260
005522' 000100  020000                  MWORD   <CJPP,J=2>                 ; pop stack     (1)
005523' 000000  000260
005524' 000200  030000                  MWORD   <CJPP,J=3>                 ; pop stack     (2)
005525' 000000  000260
005526' 000300  040000                  MWORD   <CJPP,J=4>                 ; pop stack     (3)
005527' 000000  000260
005530' 000400  050000                  MWORD   <CJPP,J=5>                 ; pop stack     (4)
005531' 000000  000260
005532' 000500  770000                  MWORD   <PUSH,J=77>                ; push stack    (5)
005533' 000000  000100
005534' 000600  770000                  MWORD   <PUSH,J=77>                ; push stack    (6)
005535' 000000  000100
005536' 000701  000000                  MWORD   <CJS,J=100>                ; push/gosub    (7)
005537' 000000  000020
005540' 001000  000000                  MWORD   <CRTN>                     ; return to 10  (10)
005541' 000000  000240

005542' 002000  210000                  MWORD   <ADDR=20,CJPP,J=21>        ; pop stack     (20)
005543' 000000  000260
005544' 002100  220000                  MWORD   <CJPP,J=22>                ; pop stack     (21)
005545' 000000  000260
005546' 002200  230000                  MWORD   <CJPP,J=23>                ; pop stack     (22)
005547' 000000  000260
005550' 002300  240000                  MWORD   <CJPP,J=24>                ; pop stack     (23)
005551' 000000  000260
005552' 002400  250000                  MWORD   <CJPP,J=25>                ; pop stack     (24)
005553' 000000  000260
005554' 002500  770000                  MWORD   <PUSH,J=77>                ; push stack    (25)
005555' 000000  000100
005556' 002600  770000                  MWORD   <PUSH,J=77>                ; push stack    (26)
005557' 000000  000100
005560' 002701  020000                  MWORD   <CJS,J=102>                ; push/gosub    (27)
005561' 000000  000020
005562' 003000  000000                  MWORD   <CRTN>                     ; return to 20  (30)
005563' 000000  000240

005564' 004000  410000                  MWORD   <ADDR=40,CJPP,J=41>        ; pop stack     (40)
005565' 000000  000260
005566' 004100  420000                  MWORD   <CJPP,J=42>                ; pop stack     (41)
005567' 000000  000260
005570' 004200  430000                  MWORD   <CJPP,J=43>                ; pop stack     (42)
005571' 000000  000260
005572' 004300  440000                  MWORD   <CJPP,J=44>                ; pop stack     (43)
005573' 000000  000260
005574' 004400  450000                  MWORD   <CJPP,J=45>                ; pop stack     (44)
005575' 000000  000260
005576' 004500  770000                  MWORD   <PUSH,J=77>                ; push stack    (45)
005577' 000000  000100
005600' 004600  770000                  MWORD   <PUSH,J=77>                ; push stack    (46)
005601' 000000  000100
```

```
  5022  005602' 004701  040000            MWORD   <CJS,J=104>                 ; push/gosub    (47)
  5023  005603' 000000  000020
  5024  005604' 005000  000000            MWORD   <CRTN>                      ; return to 50  (50)
  5025  005605' 000000  000240
  5026
  5027  005606' 006000  610000            MWORD   <ADDR=60,CJPP,J=61>         ; pop stack     (60)
  5028  005607' 000000  000260
  5029  005610' 006100  620000            MWORD   <CJPP,J=62>                 ; pop stack     (61)
  5030  005611' 000000  000260
  5031  005612' 006200  630000            MWORD   <CJPP,J=63>                 ; pop stack     (62)
  5032  005613' 000000  000260
  5033  005614' 006300  640000            MWORD   <CJPP,J=64>                 ; pop stack     (63)
  5034  005615' 000000  000260
  5035  005616' 006400  650000            MWORD   <CJPP,J=65>                 ; pop stack     (64)
  5036  005617' 000000  000260
  5037  005620' 006500  770000            MWORD   <PUSH,J=77>                 ; push stack    (65)
  5038  005621' 000000  000100
  5039  005622' 006600  770000            MWORD   <PUSH,J=77>                 ; push stack    (66)
  5040  005623' 000000  000100
  5041  005624' 006701  060000            MWORD   <CJS,J=106>                 ; push/gosub    (67)
  5042  005625' 000000  000020
  5043  005626' 007000  000000            MWORD   <CRTN>                      ; return to 70  (70)
  5044  005627' 000000  000240
  5045
  5046  005630' 010000  770000            MWORD   <ADDR=100,RFCT,DISA,CCOF,J=77>  ; return to stack addr  (100)
  5047  005631' 000000  010200
  5048  005632' 010101  010000            MWORD   <JMAP,J=101>                ; error if here         (101)
  5049  005633' 000000  000040
  5050  005634' 010200  770000            MWORD   <RFCT,DISA,CCON,J=77>       ; return to stack addr  (102)
  5051  005635' 000000  030200
  5052  005636' 010301  030000            MWORD   <JMAP,J=103>                ; error if here         (103)
  5053  005637' 000000  000040
  5054  005640' 010400  770000            MWORD   <RFCT,ENA,CCOF,J=77>        ; return to stack addr  (104)
  5055  005641' 000400  010200
  5056  005642' 010501  050000            MWORD   <JMAP,J=105>                ; error if here         (105)
  5057  005643' 000000  000040
  5058  005644' 010600  770000            MWORD   <RFCT,ENA,CCON,J=77>        ; return to stack addr  (106)
  5059  005645' 000400  030200
  5060  005646' 010701  070000            MWORD   <JMAP,J=107>                ; error if here         (107)
  5061  005647' 000000  000040
  5062
  5063  005650' 007700  770000            MWORD   <ADDR=77,JMAP,J=77>         ; error if here         (77)
  5064  005651' 000000  000040
  5065  005652' 777777  777777            -1
```

```

                                   ;#*******************************************************************
                                   ;* Test 53 - RFCT Instruction
                                   ;
                                   ; Description:  Verify  that   the   RFCT  instruction  correctly
                                   ;               decrements the counter if  nonzero, regardless of
                                   ;               the TEST inputs.
                                   ;
                                   ; Procedure:    Clear Port
                                   ;               Load test microcode (except location 50)
                                   ;
                                   ;               Load location 50 with first contents
                                   ;               Set up RAR to start at location 0, execute JMAP
                                   ;               Set MPRUN/SINGLE STEP (execute 26 LDCT,CJS,RFCT,
                                   ;                 JRP,JMAP segments)
                                   ;               Read LAR, and verify address = 616
                                   ;
                                   ;               Load location 50 with the 2nd contents
                                   ;               Set up RAR to start at location 0, execute JMAP
                                   ;               Set MPRUN/SINGLE STEP (execute 26 LDCT,CJS,RFCT,
                                   ;                 JRP segments)
                                   ;               Read LAR, and verify address = 616
                                   ;
                                   ;               Load location 50 with the 3rd contents
                                   ;               Set up RAR to start at location 0, execute JMAP
                                   ;               Set MPRUN/SINGLE STEP (execute 26 LDCT,CJS,RFCT,
                                   ;                 JRP segments)
                                   ;               Read LAR, and verify address = 616
                                   ;
                                   ;               Load location 50 with the 4th contents
                                   ;               Set up RAR to start at location 0, execute JMAP
                                   ;               Set MPRUN/SINGLE STEP (execute 26 LDCT,CJS,RFCT,
                                   ;                 JRP segments)
                                   ;               Read LAR, and verify address = 616
                                   ;
                                   ; Failure:      ---
                                   ;#*******************************************************************

                                   ; Test data

005653' 254 00 0 00 005664'        TSTS53: JRST    TG53                    ; go start test
005654' 240401  000053                     SEQ!MPROC!NDMP!ZSEQ!53          ; test mask
005655' 005 44' 011166'                    T53M,,[ASCIZ ^RFCT Instruction^]
005656' 010442' 010517'                    [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
005657' 000000  006275'                    TSTS54                          ; failure test table
005660' 000000  006377'                    TSTS55                          ;   ...
005661' 000000  006707'                    TSTS56
005662' 000000  006777'                    TSTS57
005663' 777777  777777                     -1
```

```
                                        ; Start test

005664' 201 00 0 00 000000'             TG53:   MOVEI   Z3                      ; get address of module start
005665' 260 17 0 00 005470*                     GO      TRACE                   ; handle trace output
005666' 201 01 0 00 005744'                     MOVEI   1,T53M                  ; set up microcode address
005667' 260 17 0 00 005472*                     GO      TLOAD                   ; load/verify it
005670' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

005671' 260 17 0 00 005474*             TL53:   GO      IPACLR                  ; clear port
005672' 402 00 0 00 005475*                     SETZM   TSTSUB                  ; initialize subtest number
005673' 201 06 0 00 005707'                     MOVEI   6,TS53                  ; get sstep table address
005674' 474 07 0 00 000000                      SETO    7,                      ; initialize location 50 pointer

                                        ; Loop on single step table entries

005675' 400 15 0 00 000000              TA53:   SETZ    ERFLG,                  ; clear error flag
005676' 260 17 0 00 005500*                     GO      SEXEC                   ; execute table entry
005677' 254 00 0 00 005706'                     JRST    TX53                    ; end of sstep table
005700' 254 00 0 00 005675'                     JRST    TA53                    ; keep looping after call
005701' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - end address incorrect

                                        ; Handle error printouts and scope looping

005702' 027 00 0 00 005742'                     SCOPER  MA53                    ; print error message
005703' 254 00 0 00 005671'                     JRST    TL53                    ; loop on error
005704' 254 00 0 00 005706'                     JRST    TX53                    ; altmode exit
005705' 322 15 0 00 005675'                     JUMPE   ERFLG,TA53              ; do next sstep table entry

                                        ; End of test

005706' 263 17 0 00 000000              TX53:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

005707' 300000  005720'                 TS53:   STABLE  (SSCALL,T53LOD)
005710' 120304  760616                          STABLE  (SSSTRT,^D131,476,616)
005711' 300000  005720'                         STABLE  (SSCALL,T53LOD)
005712' 120304  760616                          STABLE  (SSSTRT,^D131,476,616)
005713' 300000  005720'                         STABLE  (SSCALL,T53LOD)
005714' 120304  760616                          STABLE  (SSSTRT,^D131,476,616)
005715' 300000  005720'                         STABLE  (SSCALL,T53LOD)
005716' 120304  760616                          STABLE  (SSSTRT,^D131,476,616)
005717' 000000  000000                          STABLE  (SSLAST)
```

```

                                          ; Load location 50 with special microword

005720' 350 01 0 00 000007        T53LOD: AOS     1,7                   ; increment/get offset pointer
005721' 221 01 0 00 000003                IMULI   1,3                   ; multiply by 2
005722' 271 01 0 00 005726'               ADDI    1,T53MM               ; set up microcode address
005723' 260 17 0 00 005667*               GO      TLOAD                 ; load/verify it
005724' 255 00 0 00 000000                JFCL                          ; error - ignore
005725' 263 17 0 00 000000                RTN                           ; exit

                                          ; Special microcode for location 50

005726' 005000  510000            T53MM:  MWORD   <ADDR=50,RFCT,DISA,CCOF,J=51>   ; decrement cntr        (50)
005727' 000000  010200
005730' 777777  777777                    -1
005731' 005000  510000                    MWORD   <ADDR=50,RFCT,DISA,CCON,J=51>   ; decrement cntr        (50)
005732' 000000  030200
005733' 777777  777777                    -1
005734' 005000  510000                    MWORD   <ADDR=50,RFCT,ENA,CCOF,J=51>    ; decrement cntr        (50)
005735' 000400  010200
005736' 777777  777777                    -1
005737' 005000  510000                    MWORD   <ADDR=50,RFCT,ENA,CCON,J=51>    ; decrement cntr        (50)
005740' 000400  030200
005741' 777777  777777                    -1

                                          ; Error messages

005742' 140000  011224'           MA53:   MSG!TXNOT![ASCIZ /RFCT not decrementing cntr properly/]
005743' 000000000000#                     LAST!CALL!TXALL!SSPNT         ; print sequencer data

                                          ; Test microcode

005744' 047605  000000            T53M:   MWORD   <ADDR=476,JMAP,J=500>           ; init PC               (476)
005745' 000000  000040
005746' 047704  770000                    MWORD   <ADDR=477,JMAP,J=477>           ; error if here         (477)
005747' 000000  000040

005750' 050000  010000                    MWORD   <ADDR=500,LDCT,J=1>             ; load counter          (500)
005751' 000000  000300
005752' 050100  500000                    MWORD   <CJS,J=50>                      ; go decrement cntr     (501)
005753' 000000  000020
005754' 050200  510000                    MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (502)
005755' 000400  010160

005756' 050300  020000                    MWORD   <LDCT,J=2>                      ; load counter          (503)
005757' 000000  000300
005760' 050400  500000                    MWORD   <CJS,J=50>                      ; go decrement cntr     (504)
005761' 000000  000020
005762' 050500  510000                    MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (505)
005763' 000400  010160

005764' 050600  040000                    MWORD   <LDCT,J=4>                      ; load counter          (506)
005765' 000000  000300
005766' 050700  500000                    MWORD   <CJS,J=50>                      ; go decrement cntr     (507)
005767' 000000  000020
```

```
  5217  005770' 051000  510000                    MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (510)
  5218  005771' 000400  010160
  5219
  5220  005772' 051100  100000                    MWORD   <LDCT,J=10>                     ; load counter          (511)
  5221  005773' 000000  000300
  5222  005774' 051200  500000                    MWORD   <CJS,J=50>                      ; go decrement cntr     (512)
  5223  005775' 000000  000020
  5224  005776' 051300  510000                    MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (513)
  5225  005777' 000400  010160
  5226
  5227  006000' 051400  200000                    MWORD   <LDCT,J=20>                     ; load counter          (514)
  5228  006001' 000000  000300
  5229  006002' 051500  500000                    MWORD   <CJS,J=50>                      ; go decrement cntr     (515)
  5230  006003' 000000  000020
  5231  006004' 051600  510000                    MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (516)
  5232  006005' 000400  010160
  5233
  5234  006006' 051700  400000                    MWORD   <LDCT,J=40>                     ; load counter          (517)
  5235  006007' 000000  000300
  5236  006010' 052000  500000                    MWORD   <CJS,J=50>                      ; go decrement cntr     (520)
  5237  006011' 000000  000020
  5238  006012' 052100  510000                    MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (521)
  5239  006013' 000400  010160
  5240
  5241  006014' 052201  000000                    MWORD   <LDCT,J=100>                    ; load counter          (522)
  5242  006015' 000000  000300
  5243  006016' 052300  500000                    MWORD   <CJS,J=50>                      ; go decrement cntr     (523)
  5244  006017' 000000  000020
  5245  006020' 052400  510000                    MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (524)
  5246  006021' 000400  010160
  5247
  5248  006022' 052502  000000                    MWORD   <LDCT,J=200>                    ; load counter          (525)
  5249  006023' 000000  000300
  5250  006024' 052600  500000                    MWORD   <CJS,J=50>                      ; go decrement cntr     (526)
  5251  006025' 000000  000020
  5252  006026' 052700  510000                    MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (527)
  5253  006027' 000400  010160
  5254
  5255  006030' 053004  000000                    MWORD   <LDCT,J=400>                    ; load counter          (530)
  5256  006031' 000000  000300
  5257  006032' 053100  500000                    MWORD   <CJS,J=50>                      ; go decrement cntr     (531)
  5258  006033' 000000  000020
  5259  006034' 053200  510000                    MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (532)
  5260  006035' 000400  010160
  5261
  5262  006036' 053310  000000                    MWORD   <LDCT,J=1000>                   ; load counter          (533)
  5263  006037' 000000  000300
  5264  006040' 053400  500000                    MWORD   <CJS,J=50>                      ; go decrement cntr     (534)
  5265  006041' 000000  000020
  5266  006042' 053500  510000                    MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (535)
  5267  006043' 000400  010160
  5268
  5269  006044' 053617  770000                    MWORD   <LDCT,J=1777>                   ; load counter          (536)
  5270  006045' 000000  000300
  5271  006046' 053700  500000                    MWORD   <CJS,J=50>                      ; go decrement cntr     (537)
```

```
5272 006047' 000000 000020
5273 006050' 054000 510000              MWORD   <JRP,ENA,CCOF,J=51>           ; go to cntr addr       (540)
5274 006051' 000400 010160
5275
5276 006052' 054120 000000              MWORD   <LDCT,J=2000>                 ; load counter          (541)
5277 006053' 000000 000300
5278 006054' 054200 500000              MWORD   <CJS,J=50>                    ; go decrement cntr     (542)
5279 006055' 000000 000020
5280 006056' 054300 510000              MWORD   <JRP,ENA,CCOF,J=51>           ; go to cntr addr       (543)
5281 006057' 000400 010160
5282
5283 006060' 054427 770000              MWORD   <LDCT,J=2777>                 ; load counter          (544)
5284 006061' 000000 000300
5285 006062' 054500 500000              MWORD   <CJS,J=50>                    ; go decrement cntr     (545)
5286 006063' 000000 000020
5287 006064' 054600 510000              MWORD   <JRP,ENA,CCOF,J=51>           ; go to cntr addr       (546)
5288 006065' 000400 010160
5289
5290 006066' 054737 770000              MWORD   <LDCT,J=3777>                 ; load counter          (547)
5291 006067' 000000 000300
5292 006070' 055000 500000              MWORD   <CJS,J=50>                    ; go decrement cntr     (550)
5293 006071' 000000 000020
5294 006072' 055100 510000              MWORD   <JRP,ENA,CCOF,J=51>           ; go to cntr addr       (551)
5295 006073' 000400 010160
5296
5297 006074' 055240 000000              MWORD   <LDCT,J=4000>                 ; load counter          (552)
5298 006075' 000000 000300
5299 006076' 055300 500000              MWORD   <CJS,J=50>                    ; go decrement cntr     (553)
5300 006077' 000000 000020
5301 006100' 055400 510000              MWORD   <JRP,ENA,CCOF,J=51>           ; go to cntr addr       (554)
5302 006101' 000400 010160
5303
5304 006102' 055547 770000              MWORD   <LDCT,J=4777>                 ; load counter          (555)
5305 006103' 000000 000300
5306 006104' 055600 500000              MWORD   <CJS,J=50>                    ; go decrement cntr     (556)
5307 006105' 000000 000020
5308 006106' 055700 510000              MWORD   <JRP,ENA,CCOF,J=51>           ; go to cntr addr       (557)
5309 006107' 000400 010160
5310
5311 006110' 056053 770000              MWORD   <LDCT,J=5377>                 ; load counter          (560)
5312 006111' 000000 000300
5313 006112' 056100 500000              MWORD   <CJS,J=50>                    ; go decrement cntr     (561)
5314 006113' 000000 000020
5315 006114' 056200 510000              MWORD   <JRP,ENA.CCOF,J=51>           ; go to cntr addr       (562)
5316 006115' 000400 010160
5317
5318 006116' 056355 770000              MWORD   <LDCT,J=5577>                 ; load counter          (563)
5319 006117' 000000 000300
5320 006120' 056400 500000              MWORD   <CJS,J=50>                    ; go decrement cntr     (564)
5321 006121' 000000 000020
5322 006122' 056500 510000              MWORD   <JRP,ENA,CCOF,J=51>           ; go to cntr addr       (565)
5323 006123' 000400 010160
5324
5325 006124' 056656 770000              MWORD   <LDCT,J=5677>                 ; load counter          (566)
5326 006125' 000000 000300
```

```
006126' 056700  500000          MWORD   <CJS,J=50>                      ; go decrement cntr     (567)
006127' 000000  000020
006130' 057000  510000          MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (570)
006131' 000400  010160

006132' 057157  370000          MWORD   <LDCT,J=5737>                   ; load counter          (571)
006133' 000000  000300
006134' 057200  500000          MWORD   <CJS,J=50>                      ; go decrement cntr     (572)
006135' 000000  000020
006136' 057300  510000          MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (573)
006137' 000400  010160

006140' 057457  570000          MWORD   <LDCT,J=5757>                   ; load counter          (574)
006141' 000000  000300
006142' 057500  500000          MWORD   <CJS,J=50>                      ; go decrement cntr     (575)
006143' 000000  000020
006144' 057600  510000          MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (576)
006145' 000400  010160

006146' 057757  670000          MWORD   <LDCT,J=5767>                   ; load counter          (577)
006147' 000000  000300
006150' 060000  500000          MWORD   <CJS,J=50>                      ; go decrement cntr     (600)
006151' 000000  000020
006152' 060100  510000          MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (601)
006153' 000400  010160

006154' 060257  730000          MWORD   <LDCT,J=5773>                   ; load counter          (602)
006155' 000000  000300
006156' 060300  500000          MWORD   <CJS,J=50>                      ; go decrement cntr     (603)
006157' 000000  000020
006160' 060400  510000          MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (604)
006161' 000400  010160

006162' 060557  750000          MWORD   <LDCT,J=5775>                   ; load counter          (605)
006163' 000000  000300
006164' 060600  500000          MWORD   <CJS,J=50>                      ; go decrement cntr     (606)
006165' 000000  000020
006166' 060700  510000          MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (607)
006167' 000400  010160

006170' 061057  760000          MWORD   <LDCT,J=5776>                   ; load counter          (610)
006171' 000000  000300
006172' 061100  500000          MWORD   <CJS,J=50>                      ; go decrement cntr     (611)
006173' 000000  000020
006174' 061200  510000          MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (612)
006175' 000400  010160

006176' 061357  770000          MWORD   <LDCT,J=5777>                   ; load counter          (613)
006177' 000000  000300
006200' 061400  500000          MWORD   <CJS,J=50>                      ; go decrement cntr     (614)
006201' 000000  000020
006202' 061500  510000          MWORD   <JRP,ENA,CCOF,J=51>             ; go to cntr addr       (615)
006203' 000400  010160

006204' 061606  170000          MWORD   <JMAP,J=617>                    ; done                  (616)
```

```
5382  006205' 000000  000040
5383  006206' 061706  170000          MWORD   <JMAP,J=617>                  ; error                (617)
5384  006207' 000000  000040
5385
5386  006210' 000005  030000          MWORD   <ADDR=0,JMAP,J=503>           ; do next segment      (0)
5387  006211' 000000  000040
5388  006212' 000105  060000          MWORD   <ADDR=1,JMAP,J=506>           ; do next segment      (1)
5389  006213' 000000  000040
5390  006214' 000305  110000          MWORD   <ADDR=3,JMAP,J=511>           ; do next segment      (3)
5391  006215' 000000  000040
5392  006216' 000705  140000          MWORD   <ADDR=7,JMAP,J=514>           ; do next segment      (7)
5393  006217' 000000  000040
5394  006220' 001705  170000          MWORD   <ADDR=17,JMAP,J=517>          ; do next segment      (17)
5395  006221' 000000  000040
5396  006222' 003705  220000          MWORD   <ADDR=37,JMAP,J=522>          ; do next segment      (37)
5397  006223' 000000  000040
5398  006224' 007705  250000          MWORD   <ADDR=77,JMAP,J=525>          ; do next segment      (77)
5399  006225' 000000  000040
5400  006226' 017705  300000          MWORD   <ADDR=177,JMAP,J=530>         ; do next segment      (177)
5401  006227' 000000  000040
5402  006230' 037705  330000          MWORD   <ADDR=377,JMAP,J=533>         ; do next segment      (377)
5403  006231' 000000  000040
5404  006232' 077705  360000          MWORD   <ADDR=777,JMAP,J=536>         ; do next segment      (777)
5405  006233' 000000  000040
5406  006234' 177605  410000          MWORD   <ADDR=1776,JMAP,J=541>        ; do next segment      (1776)
5407  006235' 000000  000040
5408  006236' 177705  440000          MWORD   <ADDR=1777,JMAP,J=544>        ; do next segment      (1777)
5409  006237' 000000  000040
5410  006240' 277605  470000          MWORD   <ADDR=2776,JMAP,J=547>        ; do next segment      (2776)
5411  006241' 000000  000040
5412  006242' 377605  520000          MWORD   <ADDR=3776,JMAP,J=552>        ; do next segment      (3776)
5413  006243' 000000  000040
5414  006244' 377705  550000          MWORD   <ADDR=3777,JMAP,J=555>        ; do next segment      (3777)
5415  006245' 000000  000040
5416  006246' 477605  600000          MWORD   <ADDR=4776,JMAP,J=560>        ; do next segment      (4776)
5417  006247' 000000  000040
5418  006250' 537605  630000          MWORD   <ADDR=5376,JMAP,J=563>        ; do next segment      (5376)
5419  006251' 000000  000040
5420  006252' 557605  660000          MWORD   <ADDR=5576,JMAP,J=566>        ; do next segment      (5576)
5421  006253' 000000  000040
5422  006254' 567605  710000          MWORD   <ADDR=5676,JMAP,J=571>        ; do next segment      (5676)
5423  006255' 000000  000040
5424  006256' 573605  740000          MWORD   <ADDR=5736,JMAP,J=574>        ; do next segment      (5736)
5425  006257' 000000  000040
5426  006260' 575605  770000          MWORD   <ADDR=5756,JMAP,J=577>        ; do next segment      (5756)
5427  006261' 000000  000040
5428  006262' 576606  020000          MWORD   <ADDR=5766,JMAP,J=602>        ; do next segment      (5766)
5429  006263' 000000  000040
5430  006264' 577206  050000          MWORD   <ADDR=5772,JMAP,J=605>        ; do next segment      (5772)
5431  006265' 000000  000040
5432  006266' 577406  100000          MWORD   <ADDR=5774,JMAP,J=610>        ; do next segment      (5774)
5433  006267' 000000  000040
5434  006270' 577506  130000          MWORD   <ADDR=5775,JMAP,J=613>        ; do next segment      (5775)
5435  006271' 000000  000040
5436  006272' 577606  160000          MWORD   <ADDR=5776,JMAP,J=616>        ; do next segment      (5776)
```

```
  5437  006273' 000000  000040
  5438  006274' 777777  777777                    -1
```

```

                                      ;#*********************************************************************
                                      ;* Test 54 - RPCT Instruction
                                      ;
                                      ; Description:  Verify that the RPCT instruction causes a jump to
                                      ;               the address specified at the direct inputs if the
                                      ;               counter is nonzero.
                                      ;
                                      ;               And verify that the RPCT instruction continues to
                                      ;               the next instruction if the counter is zero.
                                      ;
                                      ;               And verify that the counter is not changed if the
                                      ;               counter is zero.
                                      ;
                                      ; Procedure:    Clear Port
                                      ;               Load test microcode
                                      ;               Set up RAR to start at location 100
                                      ;               Set MPRUN/SINGLE STEP (execute JZ,LDCT,4 RPCT's)
                                      ;               Set MPRUN/SINGLE STEP (execute LDCT,4 RPCT's)
                                      ;               Set MPRUN/SINGLE STEP (execute JRP)
                                      ;               Read LAR, and verify address = 0
                                      ;
                                      ; Failure:      ---
                                      ;#*********************************************************************

                                      ; Test data

006275' 254 00 0 00 006306'           TSTS54: JRST    TG54                          ; go start test
006276' 240401  000054                        SEQ!MPROC!NDMP!ZSEQ!54                ; test mask
006277' 006334' 011234'                       T54M,,[ASCIZ ^RPCT Instruction^]
006300' 011442' 010517'                       [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
006301' 000000  006377'                       TSTS55                                ; failure test table
006302' 000000  006707'                       TSTS56                                ;   ...
006303' 000000  006777'                       TSTS57
006304' 000000  007051'                       TSTS60
006305' 777777  777777                        -1

                                      ; Start test

006306' 201 00 0 00 000000'           TG54:   MOVEI   Z3                            ; get address of module start
006307' 260 17 0 00 005665*                   GO      TRACE                         ; handle trace output
006310' 201 01 0 00 006334'                   MOVEI   1,T54M                        ; set up microcode address
006311' 260 17 0 00 005723*                   GO      TLOAD                         ; load/verify it
006312' 263 17 0 00 000000                    RTN                                   ; failed - exit test

                                      ; Initialization

006313' 260 17 0 00 005671*           TL54:   GO      IPACLR                        ; clear port
006314' 402 00 0 00 005672*                   SETZM   TSTSUB                        ; initialize subtest number
006315' 201 06 0 00 006330'                   MOVEI   6,TS54                        ; get sstep table address
```

```

                                 ; Loop on single step table entries

006316' 400 15 0 00 000000       TA54:   SETZ    ERFLG,                  ; clear error flag
006317' 260 17 0 00 005676*              GO      SEXEC                   ; execute table entry
006320' 254 00 0 00 006327'              JRST    TX54                    ; end of sstep table
006321' 254 00 0 00 006316'              JRST    TA54                    ; keep looping after call
006322' 474 15 0 00 000000               SETO    ERFLG,                  ; error - end address incorrect

                                 ; Handle error printouts and scope looping

006323' 027 00 0 00 006332'              SCOPER  MA54                    ; print error message
006324' 254 00 0 00 006313'              JRST    TL54                    ; loop on error
006325' 254 00 0 00 006327'              JRST    TX54                    ; altmode exit
006326' 322 15 0 00 006316'              JUMPE   ERFLG,TA54              ; do next sstep table entry

                                 ; End of test

006327' 263 17 0 00 000000       TX54:   RTN                             ; return

                                 ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

006330' 101401  000100           TS54:   STABLE  (SSSTRT,^D12,100,0)
006331' 000000  000000                   STABLE  (SSLAST)

                                 ; Error messages

006332' 140000  011240'          MA54:   MSG!TXNOT![ASCIZ /RPCT did not dispatch correctly/]
006333' 000000000000#                    LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```


                                        ; Test microcode

006334' 010000  000000                  T54M:   MWORD   <ADDR=100,JZ>                   ; init PC               (100)
006335' 000000  000000
006336' 000007  770000                          MWORD   <ADDR=0,LDCT,J=777>             ; load counter          (0)
006337' 000000  000300
006340' 000100  030000                          MWORD   <RPCT,DISA,CCOF,J=3>            ; dec cntr/go to 4      (1)
006341' 000000  010220
006342' 000200  020000                          MWORD   <JMAP,J=2>                      ; error if here         (2)
006343' 000000  000040
006344' 000300  050000                          MWORD   <RPCT,DISA,CCON,J=5>            ; dec cntr/go to 6      (3)
006345' 000000  030220
006346' 000400  040000                          MWORD   <JMAP,J=4>                      ; error if here         (4)
006347' 000000  000040
006350' 000500  070000                          MWORD   <RPCT,ENA,CCOF,J=7>             ; dec cntr/go to 10     (5)
006351' 000400  010220
006352' 000600  060000                          MWORD   <JMAP,J=6>                      ; error if here         (6)
006353' 000000  000040
006354' 000700  110000                          MWORD   <RPCT,ENA,CCON,J=11>            ; dec cntr/go to 12     (7)
006355' 000400  030220
006356' 001000  100000                          MWORD   <JMAP,J=10>                     ; error if here         (10)
006357' 000000  000040

006360' 001100  000000                          MWORD   <LDCT,J=0>                      ; load counter          (11)
006361' 000000  000300
006362' 001200  770000                          MWORD   <RPCT,DISA,CCOF,J=77>           ; continue              (12)
006363' 000000  010220
006364' 001300  770000                          MWORD   <RPCT,DISA,CCOF,J=77>           ; continue              (13)
006365' 000000  010220
006366' 001400  770000                          MWORD   <RPCT,DISA,CCOF,J=77>           ; continue              (14)
006367' 000000  010220
006370' 001500  770000                          MWORD   <RPCT,DISA,CCOF,J=77>           ; continue              (15)
006371' 000000  010220
006372' 001600  770000                          MWORD   <JRP,ENA,CCOF,J=77>             ; go to cntr addr (0)   (16)
006373' 000400  010160
006374' 007700  770000                          MWORD   <ADDR=77,JMAP,J=77>             ; error if here         (77)
006375' 000000  000040
006376' 777777  777777                          -1
```

```

                                        ;#*********************************************************************
                                        ;* Test 55 - RPCT Instruction
                                        ;
                                        ; Description:  Verify that the  RPCT  instruction  correctly
                                        ;               decrements the counter if  nonzero regardless of
                                        ;               the TEST inputs.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode
                                        ;               Set up RAR to start at location 333, execute JMAP
                                        ;               Set MPRUN/SINGLE STEP (execute 24 segments of
                                        ;                 LDCT,JMAP,4 RPCT's,JRP,JMAP)
                                        ;               Read LAR, and verify address = 560
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

006377' 254 00 0 00 006410'             TSTS55: JRST    TG55                            ; go start test
006400' 240401  000055                          SEQ!MPROC!NDMP!ZSEQ!55                  ; test mask
006401' 006436' 011234'                         T55M,,[ASCIZ ^RPCT Instruction^]
006402' 010442' 010517'                         [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
006403' 000000  006707'                         TSTS56                                  ; failure test table
006404' 000000  006777'                         TSTS57                                  ;   ...
006405' 000000  007051'                         TSTS60
006406' 000000  007127'                         TSTS61
006407' 777777  777777                          -1

                                        ; Start test

006410' 201 00 0 00 000000'             TG55:   MOVEI   Z3                              ; get address of module start
006411' 260 17 0 00 006307*                     GO      TRACE                           ; handle trace output
006412' 201 01 0 00 006436'                     MOVEI   1,T55M                          ; set up microcode address
006413' 260 17 0 00 006311*                     GO      TLOAD                           ; load/verify it
006414' 263 17 0 00 000000                      RTN                                     ; failed - exit test

                                        ; Initialization

006415' 260 17 0 00 006313*             TL55:   GO      IPACLR                          ; clear port
006416' 402 00 0 00 006314*                     SETZM   TSTSUB                          ; initialize subtest number
006417' 201 06 0 00 006432'                     MOVEI   6,TS55                          ; get sstep table address

                                        ; Loop on single step table entries

006420' 400 15 0 00 000000              TA55:   SETZ    ERFLG,                          ; clear error flag
006421' 260 17 0 00 006317*                     GO      SEXEC                           ; execute table entry
006422' 254 00 0 00 006431'                     JRST    TX55                            ; end of sstep table
006423' 254 00 0 00 006420'                     JRST    TA55                            ; keep looping after call
006424' 474 15 0 00 000000                      SETO    ERFLG,                          ; error - end address incorrect
```

```
                                            ; Handle error printouts and scope looping

006425' 027 00 0 00 006434'                         SCOPER  MA55                    ; print error message
006426' 254 00 0 00 006415'                         JRST    TL55                    ; loop on error
006427' 254 00 0 00 006431'                         JRST    TX55                    ; altmode exit
006430' 322 15 0 00 006420'                         JUMPE   ERFLG,TA55              ; do next sstep table entry

                                            ; End of test

006431' 263 17 0 00 000000                  TX55:   RTN                             ; return

                                            ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

006432' 130103  330560                      TS55:   STABLE  (SSSTRT,^D193,333,560)
006433' 000000  000000                              STABLE  (SSLAST)

                                            ; Error messages

006434' 140000  011247'                     MA55:   MSG!TXNOT![ASCIZ /RPCT did not decrement the cntr properly/]
006435' 000000000000#                               LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                            ; Test microcode

                                            ;       Subroutine to do the RPCT's

006436' 033305  000000                      T55M:   MWORD   <ADDR=333,JMAP,J=500>           ; init PC               (333)
006437' 000000  000040
006440' 005000  520000                              MWORD   <ADDR=50,RPCT,DISA,CCOF,J=52>   ; dec ctr/go to 52      (50)
006441' 000000  010220
006442' 005100  510000                              MWORD   <JMAP,J=51>                     ; error if here         (51)
006443' 000000  000040
006444' 005200  540000                              MWORD   <RPCT,DISA,CCON,J=54>           ; dec ctr/go to 54      (52)
006445' 000000  030220
006446' 005300  530000                              MWORD   <JMAP,J=53>                     ; error if here         (53)
006447' 000000  000040
006450' 005400  560000                              MWORD   <RPCT,ENA,CCOF,J=56>            ; dec ctr/go to 56      (54)
006451' 000400  010220
006452' 005500  550000                              MWORD   <JMAP,J=55>                     ; error if here         (55)
006453' 000000  000040
006454' 005600  600000                              MWORD   <RPCT,ENA,CCON,J=60>            ; dec ctr/go to 60      (56)
006455' 000400  030220
006456' 005700  570000                              MWORD   <JMAP,J=57>                     ; error if here         (57)
006457' 000000  000040
006460' 006000  570000                              MWORD   <JRP,ENA,CCOF,J=57>             ; go to cntr addr       (60)
006461' 000400  010160

006462' 050000  040000                              MWORD   <ADDR=500,LDCT,J=4>             ; load counter  (500)
006463' 000000  000300
006464' 050100  500000                              MWORD   <JMAP,J=50>                     ; go do RPCT's  (501)
006465' 000000  000040
006466' 050200  100000                              MWORD   <LDCT,J=10>                     ; load counter  (502)
006467' 000000  000300
006470' 050300  500000                              MWORD   <JMAP,J=50>                     ; go do RPCT's  (503)
006471' 000000  000040
```

```
 5663  006472' 050400  200000            MWORD   <LDCT,J=20>                     ; load counter   (504)
 5664  006473' 000000  000300
 5665  006474' 050500  500000            MWORD   <JMAP,J=50>                     ; go do RPCT's   (505)
 5666  006475' 000000  000040
 5667  006476' 050600  400000            MWORD   <LDCT,J=40>                     ; load counter   (506)
 5668  006477' 000000  000300
 5669  006500' 050700  500000            MWORD   <JMAP,J=50>                     ; go do RPCT's   (507)
 5670  006501' 000000  000040
 5671  006502' 051001  000000            MWORD   <LDCT,J=100>                    ; load counter   (510)
 5672  006503' 000000  000300
 5673  006504' 051100  500000            MWORD   <JMAP,J=50>                     ; go do RPCT's   (511)
 5674  006505' 000000  000040
 5675  006506' 051202  000000            MWORD   <LDCT,J=200>                    ; load counter   (512)
 5676  006507' 000000  000300
 5677  006510' 051300  500000            MWORD   <JMAP,J=50>                     ; go do RPCT's   (513)
 5678  006511' 000000  000040
 5679  006512' 051404  000000            MWORD   <LDCT,J=400>                    ; load counter   (514)
 5680  006513' 000000  000300
 5681  006514' 051500  500000            MWORD   <JMAP,J=50>                     ; go do RPCT's   (515)
 5682  006515' 000000  000040
 5683  006516' 051610  000000            MWORD   <LDCT,J=1000>                   ; load counter   (516)
 5684  006517' 000000  000300
 5685  006520' 051700  500000            MWORD   <JMAP,J=50>                     ; go do RPCT's   (517)
 5686  006521' 000000  000040
 5687  006522' 052017  770000            MWORD   <LDCT,J=1777>                   ; load counter   (520)
 5688  006523' 000000  000300
 5689  006524' 052100  500000            MWORD   <JMAP,J=50>                     ; go do RPCT's   (521)
 5690  006525' 000000  000040
 5691  006526' 052220  000000            MWORD   <LDCT,J=2000>                   ; load counter   (522)
 5692  006527' 000000  000300
 5693  006530' 052300  500000            MWORD   <JMAP,J=50>                     ; go do RPCT's   (523)
 5694  006531' 000000  000040
 5695  006532' 052427  770000            MWORD   <LDCT,J=2777>                   ; load counter   (524)
 5696  006533' 000000  000300
 5697  006534' 052500  500000            MWORD   <JMAP,J=50>                     ; go do RPCT's   (525)
 5698  006535' 000000  000040
 5699  006536' 052637  770000            MWORD   <LDCT,J=3777>                   ; load counter   (526)
 5700  006537' 000000  000300
 5701  006540' 052700  500000            MWORD   <JMAP,J=50>                     ; go do RPCT's   (527)
 5702  006541' 000000  000040
 5703  006542' 053040  000000            MWORD   <LDCT,J=4000>                   ; load counter   (530)
 5704  006543' 000000  000300
 5705  006544' 053100  500000            MWORD   <JMAP,J=50>                     ; go do RPCT's   (531)
 5706  006545' 000000  000040
 5707  006546' 053247  770000            MWORD   <LDCT,J=4777>                   ; load counter   (532)
 5708  006547' 000000  000300
 5709  006550' 053300  500000            MWORD   <JMAP,J=50>                     ; go do RPCT's   (533)
 5710  006551' 000000  000040
 5711  006552' 053453  770000            MWORD   <LDCT,J=5377>                   ; load counter   (534)
 5712  006553' 000000  000300
 5713  006554' 053500  500000            MWORD   <JMAP,J=50>                     ; go do RPCT's   (535)
 5714  006555' 000000  000040
 5715  006556' 053655  770000            MWORD   <LDCT,J=5577>                   ; load counter   (536)
 5716  006557' 000000  000300
 5717  006560' 053700  500000            MWORD   <JMAP,J=50>                     ; go do RPCT's   (537)
 5718   006561' 000000  000040
 5719   006562' 054056  770000          MWORD   <LDCT,J=5677>                   ; load counter  (540)
 5720   006563' 000000  000300
 5721   006564' 054100  500000          MWORD   <JMAP,J=50>                     ; go do RPCT's  (541)
 5722   006565' 000000  000040
 5723   006566' 054257  370000          MWORD   <LDCT,J=5737>                   ; load counter  (542)
 5724   006567' 000000  000300
 5725   006570' 054300  500000          MWORD   <JMAP,J=50>                     ; go do RPCT's  (543)
 5726   006571' 000000  000040
 5727   006572' 054457  570000          MWORD   <LDCT,J=5757>                   ; load counter  (544)
 5728   006573' 000000  000300
 5729   006574' 054500  500000          MWORD   <JMAP,J=50>                     ; go do RPCT's  (545)
 5730   006575' 000000  000040
 5731   006576' 054657  670000          MWORD   <LDCT,J=5767>                   ; load counter  (546)
 5732   006577' 000000  000300
 5733   006600' 054700  500000          MWORD   <JMAP,J=50>                     ; go do RPCT's  (547)
 5734   006601' 000000  000040
 5735   006602' 055057  730000          MWORD   <LDCT,J=5773>                   ; load counter  (550)
 5736   006603' 000000  000300
 5737   006604' 055100  500000          MWORD   <JMAP,J=50>                     ; go do RPCT's  (551)
 5738   006605' 000000  000040
 5739   006606' 055257  750000          MWORD   <LDCT,J=5775>                   ; load counter  (552)
 5740   006607' 000000  000300
 5741   006610' 055300  500000          MWORD   <JMAP,J=50>                     ; go do RPCT's  (553)
 5742   006611' 000000  000040
 5743   006612' 055457  760000          MWORD   <LDCT,J=5776>                   ; load counter  (554)
 5744   006613' 000000  000300
 5745   006614' 055500  500000          MWORD   <JMAP,J=50>                     ; go do RPCT's  (555)
 5746   006615' 000000  000040
 5747   006616' 055657  770000          MWORD   <LDCT,J=5777>                   ; load counter  (556)
 5748   006617' 000000  000300
 5749   006620' 055700  500000          MWORD   <JMAP,J=50>                     ; go do RPCT's  (557)
 5750   006621' 000000  000040
 5751
 5752   006622' 056005  610000          MWORD   <JMAP,J=561>                    ; done          (560)
 5753   006623' 000000  000040
 5754   006624' 056105  610000          MWORD   <JMAP,J=561>                    ; error         (561)
 5755   006625' 000000  000040
 5756
 5757   006626' 000005  020000          MWORD   <ADDR=0,JMAP,J=502>             ; do next segment       (0)
 5758   006627' 000000  000040
 5759   006630' 000405  040000          MWORD   <ADDR=4,JMAP,J=504>             ; do next segment       (4)
 5760   006631' 000000  000040
 5761   006632' 001405  060000          MWORD   <ADDR=14,JMAP,J=506>            ; do next segment       (14)
 5762   006633' 000000  000040
 5763   006634' 003405  100000          MWORD   <ADDR=34,JMAP,J=510>            ; do next segment       (34)
 5764   006635' 000000  000040
 5765   006636' 007405  120000          MWORD   <ADDR=74,JMAP,J=512>            ; do next segment       (74)
 5766   006637' 000000  000040
 5767   006640' 017405  140000          MWORD   <ADDR=174,JMAP,J=514>           ; do next segment       (174)
 5768   006641' 000000  000040
 5769   006642' 037405  160000          MWORD   <ADDR=374,JMAP,J=516>           ; do next segment       (374)
 5770   006643' 000000  000040
 5771   006644' 077405  200000          MWORD   <ADDR=774,JMAP,J=520>           ; do next segment       (774)
 5772   006645' 000000  000040
```

```
  5773  006646' 177305  220000                 MWORD   <ADDR=1773,JMAP,J=522>          ; do next segment        (1773)
  5774  006647' 000000  000040
  5775  006650' 177405  240000                 MWORD   <ADDR=1774,JMAP,J=524>          ; do next segment        (1774)
  5776  006651' 000000  000040
  5777  006652' 277305  260000                 MWORD   <ADDR=2773,JMAP,J=526>          ; do next segment        (2773)
  5778  006653' 000000  000040
  5779  006654' 377305  300000                 MWORD   <ADDR=3773,JMAP,J=530>          ; do next segment        (3773)
  5780  006655' 000000  000040
  5781  006656' 377405  320000                 MWORD   <ADDR=3774,JMAP,J=532>          ; do next segment        (3774)
  5782  006657' 000000  000040
  5783  006660' 477305  340000                 MWORD   <ADDR=4773,JMAP,J=534>          ; do next segment        (4773)
  5784  006661' 000000  000040
  5785  006662' 537305  360000                 MWORD   <ADDR=5373,JMAP,J=536>          ; do next segment        (5373)
  5786  006663' 000000  000040
  5787  006664' 557305  400000                 MWORD   <ADDR=5573,JMAP,J=540>          ; do next segment        (5573)
  5788  006665' 000000  000040
  5789  006666' 567305  420000                 MWORD   <ADDR=5673,JMAP,J=542>          ; do next segment        (5673)
  5790  006667' 000000  000040
  5791  006670' 573305  440000                 MWORD   <ADDR=5733,JMAP,J=544>          ; do next segment        (5733)
  5792  006671' 000000  000040
  5793  006672' 575305  460000                 MWORD   <ADDR=5753,JMAP,J=546>          ; do next segment        (5753)
  5794  006673' 000000  000040
  5795  006674' 576305  500000                 MWORD   <ADDR=5763,JMAP,J=550>          ; do next segment        (5763)
  5796  006675' 000000  000040
  5797  006676' 576705  520000                 MWORD   <ADDR=5767,JMAP,J=552>          ; do next segment        (5767)
  5798  006677' 000000  000040
  5799  006700' 577105  540000                 MWORD   <ADDR=5771,JMAP,J=554>          ; do next segment        (5771)
  5800  006701' 000000  000040
  5801  006702' 577205  560000                 MWORD   <ADDR=5772,JMAP,J=556>          ; do next segment        (5772)
  5802  006703' 000000  000040
  5803  006704' 577305  600000                 MWORD   <ADDR=5773,JMAP,J=560>          ; do next segment        (5773)
  5804  006705' 000000  000040
  5805  006706' 777777  777777                 -1
```

```
                                                ;#*********************************************************************
                                                ;* Test 56 - TWB Instruction
                                                ;
                                                ; Description:  Verify that the TWB instruction continues to  the
                                                ;               next sequential instruction if TEST is asserted.
                                                ;
                                                ; Procedure:    Clear Port
                                                ;               Load test microcode
                                                ;               Set up RAR to start at location 100
                                                ;               Set MPRUN/SINGLE STEP (execute JZ,PUSH,3 TWB's)
                                                ;               Set MPRUN/SINGLE STEP (execute PUSH,3 TWB's,JRP)
                                                ;               Read LAR, and verify address = 0
                                                ;
                                                ; Failure:      ---
                                                ;#*********************************************************************

                                                ; Test data

 006707' 254 00 0 00 006720'                    TSTS56: JRST    TG56                    ; go start test
 006710' 240401  000056                                 SEQ!MPROC!NDMP!ZSEQ!56          ; test mask
 006711' 006746' 011260'                                T56M,,[ASCIZ ^TWB Instruction^]
 006712' 010442' 010517'                                [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
 006713' 000000  006777'                                TSTS57                          ; failure test table
 006714' 000000  007051'                                TSTS60                          ;   ...
 006715' 000000  007127'                                TSTS61
 006716' 000000  007223'                                TSTS62
 006717' 777777  777777                                 -1

                                                ; Start test

 006720' 201 00 0 00 000000'                    TG56:   MOVEI   Z3                      ; get address of module start
 006721' 260 17 0 00 006411*                            GO      TRACE                   ; handle trace output
 006722' 201 01 0 00 006746'                            MOVEI   1,T56M                  ; set up microcode address
 006723' 260 17 0 00 006413*                            GO      TLOAD                   ; load/verify it
 006724' 263 17 0 00 000000                             RTN                             ; failed - exit test

                                                ; Initialization

 006725' 260 17 0 00 006415*                    TL56:   GO      IPACLR                  ; clear port
 006726' 402 00 0 00 006416*                            SETZM   TSTSUB                  ; initialize subtest number
 006727' 201 06 0 00 006742'                            MOVEI   6,TS56                  ; get sstep table address

                                                ; Loop on single step table entries

 006730' 400 15 0 00 000000                     TA56:   SETZ    ERFLG,                  ; clear error flag
 006731' 260 17 0 00 006421*                            GO      SEXEC                   ; execute table entry
 006732' 254 00 0 00 006741'                            JRST    TX56                    ; end of sstep table
 006733' 254 00 0 00 006730'                            JRST    TA56                    ; keep looping after call
 006734' 474 15 0 00 000000                             SETO    ERFLG,                  ; error - end address incorrect
```

```
                                          ; Handle error printouts and scope looping

006735' 027 00 0 00 006744'                      SCOPER  MA56                         ; print error message
006736' 254 00 0 00 006725'                      JRST    TL56                         ; loop on error
006737' 254 00 0 00 006741'                      JRST    TX56                         ; altmode exit
006740' 322 15 0 00 006730'                      JUMPE   ERFLG,TA56                   ; do next sstep table entry

                                          ; End of test

006741' 263 17 0 00 000000                TX56:  RTN                                  ; return

                                          ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

006742' 101201  000000                    TS56:  STABLE  (SSSTRT,^D10,100,0)
006743' 000000  000000                           STABLE  (SSLAST)

                                          ; Error messages

006744' 140000  011264'                   MA56:  MSG!TXNOT![ASCIZ /TWB did not continue PC+1 with test conditions met/]
C06745' 00C000000000#                            LAST!CALL!TXALL!SSPNT                ; print sequencer data

                                          ; Test microcode

006746' 010000  000000                    T56M:  MWORD   <ADDR=100,JZ>                          ; init PC              (100)
006747' 000000  000000
006750' 000007  770000                           MWORD   <ADDR=0,PUSH,J=777>                    ; push 1/load cntr     (0)
006751' 000000  000100
006752' 000100  770000                           MWORD   <TWB,DISA,CCOF,J=77>                   ; continue to 2        (1)
006753' 000000  010360
006754' 000200  770000                           MWORD   <TWB,DISA,CCON,J=77>                   ; continue to 3        (2)
006755' 000000  030360
006756' 000300  770000                           MWORD   <TWB,ENA,CCON,J=77>                    ; continue to 4        (3)
006757' 000400  030360
006760' 000400  000000                           MWORD   <PUSH,J=0>                             ; push 5/zero cntr     (4)
006761' 000000  000100
006762' 000500  770000                           MWORD   <TWB,DISA,CCOF,J=77>                   ; continue to 6        (5)
006763' 000000  010360
006764' 000600  770000                           MWORD   <TWB,DISA,CCON,J=77>                   ; continue to 7        (6)
006765' 000000  030360
006766' 000700  770000                           MWORD   <TWB,ENA,CCON,J=77>                    ; continue to 10       (7)
006767' 000400  030360
006770' 001000  000000                           MWORD   <JRP,ENA,CCOF>                         ; go to cntr addr      (10)
006771' 000400  010160

006772' 007700  770000                           MWORD   <ADDR=77,JMAP,J=77>                    ; error if here        (77)
006773' 000000  000040
006774' 077707  770000                           MWORD   <ADDR=777,JMAP,J=777>                  ; error if here        (777)
006775' 000000  000040
006776' 777777  777777                           -1
```

```
                                          ;#**************************************************************
                                          ;* Test 57 - TWB Instruction
                                          ;
                                          ; Description:  Verify that the  TWB  instruction  jumps  to  the
                                          ;               address  specified  at  the direct inputs if TEST
                                          ;               input is not satisfied and the counter = 0.
                                          ;
                                          ; Procedure:    Clear Port
                                          ;               Load test microcode
                                          ;               Set up RAR to start at location 100
                                          ;               Set MPRUN/SINGLE STEP (execute JZ,PUSH,TWB)
                                          ;               Read LAR, and verify address = 3
                                          ;
                                          ; Failure:      ---
                                          ;#**************************************************************

                                          ; Test data

006777' 254 00 0 00 007010'               TSTS57: JRST    TG57                    ; go start test
007000' 240401  000057                            SEQ!MPROC!NDMP!ZSEQ!57          ; test mask
007001' 007036' 011260'                           T57M,,[ASCIZ ^TWB Instruction^]
007002' 010442' 010517'                           [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
007003' 000000  007051'                           TSTS60                          ; failure test table
007004' 000000  007127'                           TSTS61                          ;   ...
007005' 000000  007223'                           TSTS62
007006' 000000  007323'                           TSTS63
007007' 777777  777777                            -1

                                          ; Start test

007010' 201 00 0 00 000000'               TG57:   MOVEI   Z3                      ; get address of module start
007011' 260 17 0 00 006721*                       GO      TRACE                   ; handle trace output
007012' 201 01 0 00 007036'                       MOVEI   1,T57M                  ; set up microcode address
007013' 260 17 0 00 006723*                       GO      TLOAD                   ; load/verify it
007014' 263 17 0 00 000000                        RTN                             ; failed - exit test

                                          ; Initialization

007015' 260 17 0 00 006725*               TL57:   GO      IPACLR                  ; clear port
007016' 402 00 0 00 006726*                       SETZM   TSTSUB                  ; initialize subtest number
007017' 201 06 0 00 007032'                       MOVEI   6,TS57                  ; get sstep table address

                                          ; Loop on single step table entries

007020' 400 15 0 00 000000                TA57:   SETZ    ERFLG,                  ; clear error flag
007021' 260 17 0 00 006731*                       GO      SEXEC                   ; execute table entry
007022' 254 00 0 00 007031'                       JRST    TX57                    ; end of sstep table
007023' 254 00 0 00 007020'                       JRST    TA57                    ; keep looping after call
007024' 474 15 0 00 000000                        SETO    ERFLG,                  ; error - end address incorrect
```

```

                                ; Handle error printouts and scope looping

007025' 027 00 0 00 007034'             SCOPER  MA57                    ; print error message
007026' 254 00 0 00 007015'             JRST    TL57                    ; loop on error
007027' 254 00 0 00 007031'             JRST    TX57                    ; altmode exit
007030' 322 15 0 00 007020'             JUMPE   ERFLG,TA57              ; do next sstep table entry

                                ; End of test

007031' 263 17 0 00 000000      TX57:   RTN                             ; return

                                ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

007032' 100301  000003          TS57:   STABLE  (SSSTRT,3,100,3)
007033' 000000  000000                  STABLE  (SSLAST)

                                ; Error messages

007034' 140000  011277'         MA57:   MSG!TXNOT![ASCIZ /TWB did not dispatch correctly/]
007035' 000000000000#                   LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                ; Test microcode

007036' 010000  000000          T57M:   MWORD   <ADDR=100,JZ>                   ; init PC               (100)
007037' 000000  000000
007040' 000000  000000                  MWORD   <ADDR=0,PUSH,J=0>               ; push 1/zero cntr      (0)
007041' 000000  000100
007042' 000100  030000                  MWORD   <TWB,ENA,CCOF,J=3>              ; ena/off - go to 3     (1)
007043' 000400  010360
007044' 000200  020000                  MWORD   <JMAP,J=2>                      ; error if here         (2)
007045' 000000  000040
007046' 000300  000000                  MWORD   <JMAP,J=0>                      ; done                  (3)
007047' 000000  000040
007050' 777777  777777                  -1
```

```
                                        ;#********************************************************************
                                        ;* Test 60 - TWB Instruction
                                        ;
                                        ; Description:  Verify that the  TWB  instruction  jumps  to  the
                                        ;               address specified by the top of the stack if TEST
                                        ;               inputs are not satisfied and counter is not = 0.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode
                                        ;               Set up RAR to start at location 100
                                        ;               Set MPRUN/SINGLE STEP (execute JZ,PUSH,CJS,TWB)
                                        ;               Set MPRUN/SINGLE STEP (execute JRP,CRTN)
                                        ;               Read LAR, and verify address = 2
                                        ;
                                        ; Failure:      ---
                                        ;#********************************************************************

                                        ; Test data

007051' 254 00 0 00 007062'             TSTS60: JRST    TG60                    ; go start test
007052' 240401  000060                          SEQ!MPROC!NDMP!ZSEQ!60          ; test mask
007053' 007110' 011260'                         T60M,,[ASCIZ ^TWB Instruction^]
007054' 010442' 010517'                         [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
007055' 000000  007127'                         TSTS61                          ; failure test table
007056' 000000  007223'                         TSTS62                          ;   ...
007057' 000000  007323'                         TSTS63
007060' 000000  007373'                         TSTS64
007061' 777777  777777                          -1

                                        ; Start test

007062' 201 00 0 00 000000'             TG60:   MOVEI   Z3                      ; get address of module start
007063' 260 17 0 00 007011*                     GO      TRACE                   ; handle trace output
007064' 201 01 0 00 007110'                     MOVEI   1,T60M                  ; set up microcode address
007065' 260 17 0 00 007013*                     GO      TLOAD                   ; load/verify it
007066' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

007067' 260 17 0 00 007015*             TL60:   GO      IPACLR                  ; clear port
007070' 402 00 0 00 007016*                     SETZM   TSTSUB                  ; initialize subtest number
007071' 201 06 0 00 007104'                     MOVEI   6,TS60                  ; get sstep table address

                                        ; Loop on single step table entries

007072' 400 15 0 00 000000              TA60:   SETZ    ERFLG,                  ; clear error flag
007073' 260 17 0 00 007021*                     GO      SEXEC                   ; execute table entry
007074' 254 00 0 00 007103'                     JRST    TX60                    ; end of sstep table
007075' 254 00 0 00 007072'                     JRST    TA60                    ; keep looping after call
007076' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - end address incorrect
```

```

                                    ; Handle error printouts and scope looping

007077' 027 00 0 00 007106'                 SCOPER  MA60                        ; print error message
007100' 254 00 0 00 007067'                 JRST    TL60                        ; loop on error
007101' 254 00 0 00 007103'                 JRST    TX60                        ; altmode exit
007102' 322 15 0 00 007072'                 JUMPE   ERFLG,TA60                  ; do next sstep table entry

                                    ; End of test

007103' 263 17 0 00 000000          TX60:   RTN                                 ; return

                                    ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

007104' 100601  000002              TS60:   STABLE  (SSSTRT,6,100,2)
007105' 000000  000000                      STABLE  (SSLAST)

                                    ; Error messages

007106' 140000  011306'             MA60:   MSG!TXNOT![ASCIZ /TWB did not dispatch to stack address/]
007107' 000000000000#                       LAST!CALL!TXALL!SSPNT               ; print sequencer data

                                    ; Test microcode

007110' 000000  100000              T60M:   MWORD   <ADDR=0,PUSH,J=10>               ; push 1,load ctr 10    (0)
007111' 000000  000100
007112' 000100  100000                      MWORD   <CJS,J=10>                       ; gosub 10              (1)
007113' 000000  000020
007114' 000200  000000                      MWORD   <JRP,ENA,CCOF>                   ; go to cntr addr       (2)
007115' 000400  010160
007116' 000700  000000                      MWORD   <ADDR=7,CRTN>                    ; return (to 2)         (7)
007117' 000000  000240
007120' 001000  770000                      MWORD   <TWB,ENA,CCOF,J=77>              ; return to 2 (no pop)  (10)
007121' 000400  010360
007122' 001100  110000                      MWORD   <JMAP,J=11>                      ; error if here         (11)
007123' 000000  000040

007124' 010000  000000                      MWORD   <ADDR=100,JZ>                    ; initialize stack      (100)
007125' 000000  000000
007126' 777777  777777                      -1
```

```

                                     ;#*********************************************************************
                                     ;* Test 61 - TWB Instruction
                                     ;
                                     ; Description:  Verify that the  TWB  instruction  pops  the  top
                                     ;               entry off  the  stack  when TEST is asserted  and
                                     ;               counter  is  non-zero,  also that  the counter is
                                     ;               decremented each time.
                                     ;
                                     ; Procedure:    Clear Port
                                     ;               Load test microcode
                                     ;               Set up RAR to start at location 100
                                     ;               Set MPRUN/SINGLE STEP (execute JZ,PUSH)
                                     ;               Set MPRUN/SINGLE STEP (execute CJS,CJS,CJS,CJS)
                                     ;               Set MPRUN/SINGLE STEP (execute TWB,TWB,TWB)
                                     ;               Set MPRUN/SINGLE STEP (execute CRTN,JRP)
                                     ;               Read LAR, and verify address = 774
                                     ;
                                     ; Failure:      ---
                                     ;#*********************************************************************

                                     ; Test data

007127' 254 00 0 00 007140'          TSTS61: JRST    TG61                    ; go start test
007130' 240401  000061                       SEQ!MPROC!NDMP!ZSEQ!61          ; test mask
007131' 007166' 011260'                      T61M,,[ASCIZ ^TWB Instruction^]
007132' 010442' 010517'                      [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
007133' 000000  007223'                      TSTS62                          ; failure test table
007134' 000000  007323'                      TSTS63                          ;   ...
007135' 000000  007373'                      TSTS64
007136' 000000  007543'                      TSTS65
007137' 777777  777777                       -1

                                     ; Start test

007140' 201 00 0 00 000000'          TG61:   MOVEI   Z3                      ; get address of module start
007141' 260 17 0 00 007C                     GO      TRACE                   ; handle trace output
007142' 201 01 0 00 007166'                  MOVEI   1,T61M                  ; set up microcode address
007143' 260 17 0 00 007065*                  GO      TLOAD                   ; load/verify it
007144' 263 17 0 00 000000                   RTN                             ; failed - exit test

                                     ; Initialization

007145' 260 17 0 00 007067*          TL61:   GO      IPACLR                  ; clear port
007146' 402 00 0 00 007070*                  SETZM   TSTSUB                  ; initialize subtest number
007147' 201 06 0 00 007162'                  MOVEI   6,TS61                  ; get sstep table address
```

```

                                      ; Loop on single step table entries

 007150' 400 15 0 00 000000          TA61:   SETZ    ERFLG,                  ; clear error flag
 007151' 260 17 0 00 007073*                 GO      SEXEC                   ; execute table entry
 007152' 254 00 0 00 007161'                 JRST    TX61                    ; end of sstep table
 007153' 254 00 0 00 007150'                 JRST    TA61                    ; keep looping after call
 007154' 474 15 0 00 000000                  SETO    ERFLG,                  ; error - end address incorrect

                                      ; Handle error printouts and scope looping

 007155' 027 00 0 00 007164'                 SCOPER  MA61                    ; print error message
 007156' 254 00 0 00 007145'                 JRST    TL61                    ; loop on error
 007157' 254 00 0 00 007161'                 JRST    TX61                    ; altmode exit
 007160' 322 15 0 00 007150'                 JUMPE   ERFLG,TA61              ; do next sstep table entry

                                      ; End of test

 007161' 263 17 0 00 000000          TX61:   RTN                             ; return

                                      ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

 007162' 101301  000774              TS61:   STABLE  (SSSTRT,^D11,100,774)
 007163' 000000  000000                      STABLE  (SSLAST)

                                      ; Error messages

 007164' 140000  011316'             MA61:   MSG!TXNOT![ASCIZ /TWB did not pop correctly or decrement cntr properly/]
 007165' 000000000000#                       LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```

                                        ; Test microcode

007166' 000007  770000          T61M:   MWORD   <ADDR=0,PUSH,J=777>             ; push 1,load ctr 777   (0)
007167' 000000  000100
007170' 000100  030000                  MWORD   <CJS,J=3>                       ; gosub 3               (1)
007171' 000000  000020
007172' 000200  000000                  MWORD   <JRP,ENA,CCOF>                  ; go to cntr addr       (2)
007173' 000400  010160
007174' 000300  050000                  MWORD   <CJS,J=5>                       ; gosub 5               (3)
007175' 000000  000020
007176' 000400  040000                  MWORD   <JMAP,J=4>                      ; error if here         (4)
007177' 000000  000040
007200' 000500  070000                  MWORD   <CJS,J=7>                       ; gosub 7               (5)
007201' 000000  000020
007202' 000600  040000                  MWORD   <JMAP,J=4>                      ; error if here         (6)
007203' 000000  000040
007204' 000700  110000                  MWORD   <CJS,J=11>                      ; gosub 11              (7)
007205' 000000  000020
007206' 001000  100000                  MWORD   <JMAP,J=10>                     ; error if here         (10)
007207' 000000  000040

007210' 001100  770000                  MWORD   <TWB,DISA,CCOF,J=77>            ; decr cntr/pop stack   (11)
007211' 000000  010360
007212' 001200  770000                  MWORD   <TWB,DISA,CCON,J=77>            ; decr cntr/pop stack   (12)
007213' 000000  030360
007214' 001300  770000                  MWORD   <TWB,ENA,CCON,J=77>             ; decr cntr/pop stack   (13)
007215' 000400  030360
007216' 001400  000000                  MWORD   <CRTN>                          ;                       (14)
007217' 000000  000240

007220' 010000  000000                  MWORD   <ADDR=100,JZ>                   ; initialize stack      (100)
007221' 000000  000000
007222' 777777  777777                  -1
```

```
                                   ;#*********************************************************************
                                   ;* Test 62 - TWB Instruction
                                   ;
                                   ; Description:  Verify that the  TWB  instruction  pops  the  top
                                   ;               entry off the stack when the counter is zero.
                                   ;
                                   ; Procedure:    Clear Port
                                   ;               Load test microcode
                                   ;               Set up RAR to start at location 100
                                   ;               Set MPRUN/SINGLE STEP (execute JZ,5 PUSH's)
                                   ;               Set MPRUN/SINGLE STEP (execute 4 TWB's,CRTN)
                                   ;               Read LAR, and verify address = 1
                                   ;
                                   ; Failure:      ---
                                   ;#*********************************************************************

                                   ; Test data

007223' 254 00 0 00 007234'        TSTS62: JRST    TG62                    ; go start test
007224' 240401  000062                     SEQ!MPROC!NDMP!ZSEQ!62          ; test mask
007225' 007262' 011260'                    T62M,,[ASCIZ ^TWB Instruction^]
007226' 010442' 010517'                    [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
007227' 000000  007323'                    TSTS63                          ; failure test table
007230' 000000  007373'                    TSTS64                          ;   ...
007231' 000000  007543'                    TSTS65
007232' 000000  007710'                    TSTS66
007233' 777777  777777                     -1

                                   ; Start test

007234' 201 00 0 00 000000'        TG62:   MOVEI   Z3                      ; get address of module start
007235' 260 17 0 00 007141*                GO      TRACE                   ; handle trace output
007236' 201 01 0 00 007262'                MOVEI   1,T62M                  ; set up microcode address
007237' 260 17 0 00 007143*                GO      TLOAD                   ; load/verify it
007240' 263 17 0 00 000000                 RTN                             ; failed - exit test

                                   ; Initialization

007241' 260 17 0 00 007145*        TL62:   GO      IPACLR                  ; clear port
007242' 402 00 0 00 007146*                SETZM   TSTSUB                  ; initialize subtest number
007243' 201 06 0 00 007256'                MOVEI   6,TS62                  ; get sstep table address

                                   ; Loop on single step table entries

007244' 400 15 0 00 000000         TA62:   SETZ    ERFLG,                  ; clear error flag
007245' 260 17 0 00 007151*                GO      SEXEC                   ; execute table entry
007246' 254 00 0 00 007255'                JRST    TX62                    ; end of sstep table
007247' 254 00 0 00 007244'                JRST    TA62                    ; keep looping after call
007250' 474 15 0 00 000000                 SETO    ERFLG,                  ; error - end address incorrect

                                   ; Handle error printouts and scope looping

007251' 027 00 0 00 007260'                SCOPER  MA62                    ; print error message
007252' 254 00 0 00 007241'                JRST    TL62                    ; loop on error
```

```
007253' 254 00 0 00 007255'              JRST    TX62                    ; altmode exit
007254' 322 15 0 00 007244'              JUMPE   ERFLG,TA62              ; do next sstep table entry

                                 ; End of test

007255' 263 17 0 00 000000       TX62:   RTN                             ; return

                                 ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

007256' 101301  000001           TS62:   STABLE  (SSSTRT,^D11,100,1)
007257' 000000  000000                   STABLE  (SSLAST)

                                 ; Error messages

007260' 140000  011331'          MA62:   MSG!TXNOT![ASCIZ /TWB did not do a pop when it should have/]
007261' 000000000000#                    LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                 ; Test microcode

007262' 000000  000000           T62M:   MWORD   <ADDR=0,PUSH,J=0>               ; push 1/zero cntr      (0)
007263' 000000  000100
007264' 000100  000000                   MWORD   <PUSH,J=0>                      ; push 2/zero cntr      (1)
007265' 000000  000100
007266' 000200  000000                   MWORD   <PUSH,J=0>                      ; push 3/zero cntr      (2)
007267' 000000  000100
007270' 000300  000000                   MWORD   <PUSH,J=0>                      ; push 4/zero cntr      (3)
007271' 000000  000100
007272' 000400  000000                   MWORD   <PUSH,J=0>                      ; push 5/zero cntr      (4)
007273' 000000  000100
007274' 000500  700000                   MWORD   <TWB,ENA,CCON,J=70>             ; pop stack,continue    (5)
007275' 000400  030360
007276' 000600  100000                   MWORD   <TWB,ENA,CCOF,J=10>             ; pop stack,goto 10     (6)
007277' 000400  010360
007300' 000700  710000                   MWORD   <JMAP,J=71>                     ; error if here         (7)
007301' 000000  000040
007302' 001000  720000                   MWORD   <TWB,DISA,CCOF,J=72>            ; pop stack,continue    (10)
007303' 000000  010360
007304' 001100  730000                   MWORD   <TWB,DISA,CCON,J=73>            ; pop stack,continue    (11)
007305' 000000  030360
007306' 001200  000000                   MWORD   <CRTN>                          ; return to 1           (12)
007307' 000000  000240
007310' 001300  700000                   MWORD   <JMAP,J=70>                     ; error if here         (70)
007311' 000000  000040
007312' 001400  710000                   MWORD   <JMAP,J=71>                     ; error if here         (71)
007313' 000000  000040
007314' 001500  720000                   MWORD   <JMAP,J=72>                     ; error if here         (72)
007315' 000000  000040
007316' 001600  730000                   MWORD   <JMAP,J=73>                     ; error if here         (73)
007317' 000000  000040
007320' 010000  000000                   MWORD   <ADDR=100,JZ>                   ; initialize stack      (100)
007321' 000000  000000
007322' 777777  777777                   -1
```

```
                                        ;#*******************************************************************
                                        ;* Test 63 - TWB Instruction
                                        ;
                                        ; Description:  Verify that the TWB instruction does  not pop the
                                        ;               stack when the counter is not = 0 and TEST inputs
                                        ;               are not satisfied.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode
                                        ;               Set up RAR to start at location 100
                                        ;               Set MPRUN/SINGLE STEP (execute JZ,PUSH,PUSH)
                                        ;               Set MPRUN/SINGLE STEP (execute TWB,TWB,TWB)
                                        ;               Read LAR, and verify address = 2
                                        ;
                                        ; Failure:      ---
                                        ;#*******************************************************************

                                        ; Test data

007323' 254 00 0 00 007334'             TSTS63: JRST    TG63                    ; go start test
007324' 240401  000063                          SEQ!MPRC'!NDMP!ZSEQ!63          ; test mask
007325' 007362' 011260'                         T63M,,[ASCIZ ^TWB Instruction^]
007326' 010442' 010517'                         [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
007327' 000000  007373'                         TSTS64                          ; failure test table
007330' 000000  007543'                         TSTS65                          ;   ...
007331' 000000  007710'                         TSTS66
007332' 000000  010114'                         TSTS67
007333' 777777  777777                          -1

                                        ; Start test

007334' 201 00 0 00 000000'             TG63:   MOVEI   %3                      ; get address of module start
007335' 260 17 0 00 007235*                     GO      TRACE                   ; handle trace output
007336' 201 01 0 00 007362'                     MOVEI   1,T63M                  ; set up microcode address
007337' 260 17 0 00 007237*                     GO      TLOAD                   ; load/verify it
007340' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

007341' 260 17 0 00 007241*             TL63:   GO      IPACLR                  ; clear port
007342' 402 00 0 00 007242*                     SETZM   TSTSUB                  ; initialize subtest number
007343' 201 06 0 00 007356'                     MOVEI   6,TS63                  ; get sstep table address

                                        ; Loop on single step table entries

007344' 400 15 0 00 000000              TA63:   SETZ    ERFLG,                  ; clear error flag
007345' 260 17 0 00 007245*                     GO      SEXEC                   ; execute table entry
007346' 254 00 0 00 007355'                     JRST    TX63                    ; end of sstep table
007347' 254 00 0 00 007344'                     JRST    TA63                    ; keep looping after call
007350' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - end address incorrect
```

```

                                   ; Handle error printouts and scope looping

007351' 027 00 0 00 007360'                SCOPER  MA63                    ; print error message
007352' 254 00 0 00 007341'                JRST    TL63                    ; loop on error
007353' 254 00 0 00 007355'                JRST    TX63                    ; altmode exit
007354' 322 15 0 00 007344'                JUMPE   ERFLG,TA63              ; do next sstep table entry

                                   ; End of test

007355' 263 17 0 00 000000         TX63:   RTN                             ; return

                                   ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

007356' 100601  000002             TS63:   STABLE  (SSSTRT,6,100,2)
007357' 000000  000000                     STABLE  (SSLAST)

                                   ; Error messages

007360' 140000  011342'            MA63:   MSG!TXNOT![ASCIZ /TWB popped stack when it shouldn't have/]
007361' 000000000000#                      LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                   ; Test microcode

007362' 000007  770000             T63M:   MWORD   <ADDR=0,PUSH,J=777>             ; push 1/load ctr 777   (0)
007363' 000000  000100
007364' 000107  770000                     MWORD   <PUSH,J=777>                    ; push 2/load ctr 777   (1)
007365' 000000  000100
007366' 000200  030000                     MWORD   <TWB,ENA,CCOF,J=3>              ; go to top of stack (2)(2)
007367' 000400  010360
007370' 010000  000000                     MWORD   <ADDR=100,JZ>                   ; initialize stack      (100)
007371' 000000  000000
007372' 777777  777777                     -1
```

```
                                        ;#*********************************************************************
                                        ;* Test 64 - Stack Interference Tests
                                        ;
                                        ; Description:  Verify that certain instructions  do  not  affect
                                        ;               the stack (register/counter nonzero).
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load test microcode
                                        ;               Set up RAR to start at location 100
                                        ;               Set MPRUN/SINGLE STEP 34 (decimal) times
                                        ;               Read LAR, and verify address = 2
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

007373' 254 00 0 00 007404'             TSTS64: JRST    TG64                    ; go start test
007374' 240401  000064                          SEQ!MPROC!NDMP!ZSEQ!64          ; test mask
007375' 007432' 011352'                         T64M,,[ASCIZ ^Stack Interference Tests^]
007376' 010442' 010517'                         [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
007377' 000000  007543'                         TSTS65                          ; failure test table
007400' 000000  007710'                         TSTS66                          ;   ...
007401' 000000  010114'                         TSTS67
007402' 000000  010273'                         TSTS70
007403' 777777  777777                          -1

                                        ; Start test

007404' 201 00 0 00 000000'             TG64:   MOVEI   Z3                      ; get address of module start
007405' 260 17 0 00 007335*                     GO      TRACE                   ; handle trace output
007406' 201 01 0 00 007432'                     MOVEI   1,T64M                  ; set up microcode address
007407' 260 17 0 00 007337*                     GO      TLOAD                   ; load/verify it
007410' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

007411' 260 17 0 00 007341*             TL64:   GO      IPACLR                  ; clear port
007412' 402 00 0 00 007342*                     SETZM   TSTSUB                  ; initialize subtest number
007413' 201 06 0 00 007426'                     MOVEI   6,TS64                  ; get sstep table address

                                        ; Loop on single step table entries

007414' 400 15 0 00 000000              TA64:   SETZ    ERFLG,                  ; clear error flag
007415' 260 17 0 00 007345*                     GO      SEXEC                   ; execute table entry
007416' 254 00 0 00 007425'                     JRST    TX64                    ; end of sstep table
007417' 254 00 0 00 007414'                     JRST    TA64                    ; keep looping after call
007420' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - end address incorrect
```

```

                                   ; Handle error printouts and scope looping

007421' 027 00 0 00 007430'                SCOPER  MA64                    ; print error message
007422' 254 00 0 00 007411'                JRST    TL64                    ; loop on error
007423' 254 00 0 00 007425'                JRST    TX64                    ; altmode exit
007424' 322 15 0 00 007414'                JUMPE   ERFLG,TA64              ; do next sstep table entry

                                   ; End of test

007425' 263 17 0 00 000000         TX64:   RTN                             ; return

                                   ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

007426' 104301  000002             TS64:   STABLE  (SSSTRT,^D35,100,2)
007427' 000000  000000                     STABLE  (SSLAST)

                                   ; Error messages

007430' 140000  011357'            MA64:   MSG!TXNOT![ASCIZ /Stack was affected when it shouldn't have been/]
007431' 000000000000#                      LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                   ; Test microcode

007432' 010000  000000             T64M:   MWORD   <ADDR=100,JZ>                     ; init PC      (100)
007433' 000000  000000
007434' 000007  770000                     MWORD   <ADDR=0,PUSH,ENA,CCON,J=777>      ; ena/on       (0)
007435' 000400  030100
007436' 000100  000000                     MWORD   <PUSH,ENA,CCOF>                   ; ena/off      (1)
007437' 000400  010100
007440' 000200  030000                     MWORD   <JMAP,DISA,CCOF,J=3>              ; disa/off     (2)
007441' 000000  010040
007442' 000300  040000                     MWORD   <JMAP,DISA,CCON,J=4>              ; disa/on      (3)
007443' 000000  030040
007444' 000400  050000                     MWORD   <JMAP,ENA,CCOF,J=5>               ; ena/off      (4)
007445' 000400  010040
007446' 000500  060000                     MWORD   <JMAP,ENA,CCON,J=6>               ; ena/on       (5)
007447' 000400  030040
007450' 000600  070000                     MWORD   <CJP,DISA,CCOF,J=7>               ; disa/off     (6)
007451' 000000  010060
007452' 000700  100000                     MWORD   <CJP,DISA,CCON,J=10>              ; disa/on      (7)
007453' 000000  030060
007454' 001007  770000                     MWORD   <CJP,ENA,CCOF,J=777>              ; ena/off      (10)
007455' 000400  010060
007456' 001100  120000                     MWORD   <CJP,ENA,CCON,J=12>               , ena/on       (11)
007457' 000400  030060
007460' 001200  130000                     MWORD   <CJV,DISA,CCOF,J=13>              ; disa/off     (12)
007461' 000000  010140
007462' 001300  140000                     MWORD   <CJV,DISA,CCON,J=14>              ; disa/on      (13)
007463' 000000  030140
007464' 001407  770000                     MWORD   <CJV,ENA,CCOF,J=777>              ; ena/off      (14)
007465' 000400  010140
007466' 001500  160000                     MWORD   <CJV,ENA,CCON,J=16>               ; ena/on       (15)
007467' 000400  030140
007470' 001600  170000                     MWORD   <RPCT,DISA,CCOF,J=17>             ; disa/off     (16)
```

```
6486  007471' 000000  010220
6487  007472' 001700  200000             MWORD   <RPCT,DISA,CCON,J=20>         ; disa/on        (17)
6488  007473' 000000  030220
6489  007474' 002000  210000             MWORD   <RPCT,ENA,CCOF,J=21>          ; ena/off        (20)
6490  007475' 000400  010220
6491  007476' 002100  220000             MWORD   <RPCT,ENA,CCON,J=22>          ; ena/on         (21)
6492  007477' 000400  030220
6493  007500' 002207  770000             MWORD   <CONT,DISA,CCOF,J=777>        ; disa/off       (22)
6494  007501' 000000  010340
6495  007502' 002307  770000             MWORD   <CONT,DISA,CCON,J=777>        ; disa/on        (23)
6496  007503' 000000  030340
6497  007504' 002407  770000             MWORD   <CONT,ENA,CCOF,J=777>         ; ena/off        (24)
6498  007505' 000400  010340
6499  007506' 002507  770000             MWORD   <CONT,ENA,CCON,J=777>         ; ena/on         (25)
6500  007507' 000400  030340
6501  007510' 002607  770000             MWORD   <LDCT,DISA,CCOF,J=777>        ; disa/off       (26)
6502  007511' 000000  010300
6503  007512' 002707  770000             MWORD   <LDCT,DISA,CCON,J=777>        ; disa/on        (27)
6504  007513' 000000  030300
6505  007514' 003007  770000             MWORD   <LDCT,ENA,CCOF,J=777>         ; ena/off        (30)
6506  007515' 000400  010300
6507  007516' 003100  330000             MWORD   <LDCT,ENA,CCON,J=33>          ; ena/on         (31)
6508  007517' 000400  030300
6509  007520' 003200  330000             MWORD   <JRP,ENA,CCOF,J=33>           ; ena/off        (32)
6510  007521' 000400  010160
6511  007522' 003300  340000             MWORD   <JRP,DISA,CCOF,J=34>          ; disa/off       (33)
6512  007523' 000000  010160
6513  007524' 003400  350000             MWORD   <JRP,DISA,CCON,J=35>          ; disa/on        (34)
6514  007525' 000000  030160
6515  007526' 003500  360000             MWORD   <JRP,ENA,CCON,J=36>           ; ena/on         (35)
6516  007527' 000400  030160
6517  007530' 003607  770000             MWORD   <CJS,ENA,CCOF,J=777>          ; ena/off        (36)
6518  007531' 000400  010020
6519  007532' 003707  770000             MWORD   <CRTN,ENA,CCOF,J=777>         ; ena/off        (37)
6520  007533' 000400  010240
6521  007534' 004007  770000             MWORD   <CJPP,ENA,CCOF,J=777>         ; ena/off        (40)
6522  007535' 000400  010260
6523
6524  007536' 004100  000000             MWORD   <CRTN,ENA,CCON>               ; ena/on         (41)
6525  007537' 000400  030240
6526
6527  007540' 077707  770000             MWORD   <ADDR=777,JMAP,J=777>         ; error if here (777)
6528  007541' 000000  000040
6529  007542' 777777  777777             -1
```

```
                                      ;#*****************************************************************
                                      ;* Test 65 - Stack Interference Tests
                                      ;
                                      ; Description:  Verify that certain instructions  do  not  affect
                                      ;               the stack (register/counter zero).
                                      ;
                                      ; Procedure:    Clear Port
                                      ;               Load test microcode
                                      ;               Set up RAR to start at location 100
                                      ;               Set MPRUN/SINGLE STEP 34 (decimal) times
                                      ;               Read LAR, and verify address = 2
                                      ;
                                      ; Failure:      ---
                                      ;#*****************************************************************

                                      ; Test data

007543' 254 00 0 00 007553'           TSTS65: JRST    TG65                    ; go start test
007544' 240401  000065                        SEQ!MPROC!NDMP!ZSEQ!65          ; test mask
007545' 007601' 011352'                       T65M,,[ASCIZ ^Stack Interference Tests^]
007546' 010442' 010517'                       [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
007547' 000000  007710'                       TSTS66                          ; failure test table
007550' 000000  010114'                       TSTS67                          ;   ...
007551' 000000  010273'                       TSTS70
007552' 777777  777777                        -1

                                      ; Start test

007553' 201 00 0 00 000000'           TG65:   MOVEI   Z3                      ; get address of module start
007554' 260 17 0 00 007405*                   GO      TRACE                   ; handle trace output
007555' 201 01 0 00 007601'                   MOVEI   1,T65M                  ; set up microcode address
007556' 260 17 0 00 007407*                   GO      TLOAD                   ; load/verify it
007557' 263 17 0 00 000000                    RTN                             ; failed - exit test

                                      ; Initialization

007560' 260 17 0 00 007411*           TL65:   GO      IPACLR                  ; clear port
007561' 402 00 0 00 007412*                   SETZM   TSTSUB                  ; initialize subtest number
007562' 201 06 0 00 007575'                   MOVEI   6,TS65                  ; get sstep table address

                                      ; Loop on single step table entries

007563' 400 15 0 00 000000            TA65:   SETZ    ERFLG,                  ; clear error flag
007564' 260 17 0 00 007415*                   GO      SEXEC                   ; execute table entry
007565' 254 00 0 00 007574'                   JRST    TX65                    ; end of sstep table
007566' 254 00 0 00 007563'                   JRST    TA65                    ; keep looping after call
007567' 474 15 0 00 000000                    SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

007570' 027 00 0 00 007577'                     SCOPER  MA65                    ; print error message
007571' 254 00 0 00 007560'                     JRST    TL65                    ; loop on error
007572' 254 00 0 00 007574'                     JRST    TX65                    ; altmode exit
007573' 322 15 0 00 007563'                     JUMPE   ERFLG,TA65              ; do next sstep table entry

                                        ; End of test

007574' 263 17 0 00 000000              TX65:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

007575' 104201  000002                  TS65:   STABLE  (SSSTRT,^D34,100,2)
007576' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

007577' 140000  011357'                 MA65:   MSG!TXNOT![ASCIZ /Stack was affected when it shouldn't have been/]
007600' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

007601' 010000  000000                  T65M:   MWORD   <ADDR=100,JZ>                   ; init PC       (100)
007602' 000000  000000
007603' 000000  000000                          MWORD   <ADDR=0,PUSH,ENA,CCON,J=0>      ; ena/on        (0)
007604' 000400  030100
007605' 000100  000000                          MWORD   <PUSH,ENA,CCOF>                 ; ena/off       (1)
007606' 000400  010100
007607' 000200  030000                          MWORD   <JMAP,DISA,CCOF,J=3>            ; disa/off      (2)
007610' 000000  010040
007611' 000300  040000                          MWORD   <JMAP,DISA,CCON,J=4>            ; disa/on       (3)
007612' 000000  030040
007613' 000400  050000                          MWORD   <JMAP,ENA,CCOF,J=5>             ; ena/on        (4)
007614' 000400  010040
007615' 000500  060000                          MWORD   <JMAP,ENA,CCON,J=6>             ; ena/off       (5)
007616' 000400  030040
007617' 000600  070000                          MWORD   <CJP,DISA,CCOF,J=7>             ; disa/off      (6)
007620' 000000  010060
007621' 000700  100000                          MWORD   <CJP,DISA,CCON,J=10>            ; disa/on       (7)
007622' 000000  030060
007623' 001007  770000                          MWORD   <CJP,ENA,CCOF,J=777>            ; ena/off       (10)
007624' 000400  010060
007625' 001100  120000                          MWORD   <CJP,ENA,CCON,J=12>             ; ena/on        (11)
007626' 000400  030060
007627' 001200  130000                          MWORD   <CJV,DISA,CCOF,J=13>            ; disa/off      (12)
007630' 000000  010140
007631' 001300  140000                          MWORD   <CJV,DISA,CCON,J=14>            ; disa/on       (13)
007632' 000000  030140
007633' 001407  770000                          MWORD   <CJV,ENA,CCOF,J=777>            ; ena/off       (14)
007634' 000400  010140
007635' 001500  160000                          MWORD   <CJV,ENA,CCON,J=16>             ; ena/on        (15)
007636' 000400  030140
007637' 001607  770000                          MWORD   <RPCT,DISA,CCOF,J=777>          ; disa/off      (16)
```

```
 6633  007640' 000000  010220
 6634  007641' 001707  770000            MWORD   <RPCT,DISA,CCON,J=777>          ; disa/on       (17)
 6635  007642' 000000  030220
 6636  007643' 002007  770000            MWORD   <RPCT,ENA,CCOF,J=777>           ; ena/off       (20)
 6637  007644' 000400  010220
 6638  007645' 002107  770000            MWORD   <RPCT,ENA,CCON,J=777>           ; ena/on        (21)
 6639  007646' 000400  030220
 6640  007647' 002207  770000            MWORD   <CONT,DISA,CCOF,J=777>          ; disa/off      (22)
 6641  007650' 000000  010340
 6642  007651' 002307  770000            MWORD   <CONT,DISA,CCON,J=777>          ; disa/on       (23)
 6643  007652' 000000  030340
 6644  007653' 002407  770000            MWORD   <CONT,ENA,CCOF,J=777>           ; ena/off       (24)
 6645  007654' 000400  010340
 6646  007655' 002507  770000            MWORD   <CONT,ENA,CCON,J=777>           ; ena/on        (25)
 6647  007656' 000400  030340
 6648  007657' 002600  000000            MWORD   <LDCT,DISA,CCOF,J=0>            ; disa/off      (26)
 6649  007660' 000000  010300
 6650  007661' 002700  000000            MWORD   <LDCT,DISA,CCON,J=0>            ; disa/on       (27)
 6651  007662' 000000  030300
 6652  007663' 003000  000000            MWORD   <LDCT,ENA,CCOF,J=0>             ; ena/off       (30)
 6653  007664' 000400  010300
 6654  007665' 003100  000000            MWORD   <LDCT,ENA,CCON,J=0>             ; ena/on        (31)
 6655  007666' 000400  030300
 6656  007667' 003200  330000            MWORD   <JRP,DISA,CCOF,J=33>            ; disa/off      (32)
 6657  007670' 000000  010160
 6658  007671' 003300  340000            MWORD   <JRP,DISA,CCON,J=34>            ; disa/on       (33)
 6659  007672' 000000  030160
 6660  007673' 003400  350000            MWORD   <JRP,ENA,CCON,J=35>             ; ena/on        (34)
 6661  007674' 000400  030160
 6662  007675' 003507  770000            MWORD   <CJS,ENA,CCOF,J=777>            ; ena/off       (35)
 6663  007676' 000400  010020
 6664  007677' 003607  770000            MWORD   <CRTN,ENA,CCOF,J=777>           ; ena/off       (36)
 6665  007700' 000400  010240
 6666  007701' 003707  770000            MWORD   <CJPP,ENA,CCOF,J=777>           ; ena/off       (37)
 6667  007702' 000400  010260
 6668
       007703' 004000  000000            MWORD   <CRTN,ENA,CCON>                 ; ena/on        (40)
 6670  007704' 000400  030240
 6671
 6672  007705' 077707  770000            MWORD   <ADDR=777,JMAP,J=777>           ; error if here (777)
 6673  007706' 000000  000040
 6674  007707' 777777  777777            -1
```

```
                                          ;#**********************************************************************
                                          ;* Test 66 - Reg/Ctr Interference Tests
                                          ;
                                          ; Description:  Verify that certain instructions  do  not  affect
                                          ;               the register/counter (which is initially zero).
                                          ;
                                          ; Procedure:    Clear Port
                                          ;               Load test microcode
                                          ;               Set up RAR to start at location 100
                                          ;               Set MPRUN/SINGLE STEP 50 (decimal) times
                                          ;               Read LAR, and verify address = 0
                                          ;
                                          ; Failure:      ---
                                          ;#**********************************************************************

                                          ; Test data

007710' 254 00 0 00 007717'               TSTS66: JRST    TG66                    ; go start test
007711' 240401  000066                            SEQ!MPROC!NDMP!ZSEQ!66          ; test mask
007712' 007745' 011371'                           T66M,,[ASCIZ ^Reg/Ctr Interference Tests^]
007713' 010442' 010517'                           [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
007714' 000000  010114'                           TSTS67                          ; failure test table
007715' 000000  010273'                           TSTS70                          ;   ...
007716' 777777  777777                            -1

                                          ; Start test

007717' 201 00 0 00 000000'               TG66:   MOVEI   Z3                      ; get address of module start
007720' 260 17 0 00 007554*                       GO      TRACE                   ; handle trace output
007721' 201 01 0 00 007745'                       MOVEI   1,T66M                  ; set up microcode address
007722' 260 17 0 00 007556*                       GO      TLOAD                   ; load/verify it
007723' 263 17 0 00 000000                        RTN                             ; failed - exit test

                                          ; Initialization

007724' 260 17 0 00 007560*               TL66:   GO      IPACLR                  ; clear port
007725' 402 00 0 00 007561*                       SETZM   TSTSUB                  ; initialize subtest number
007726' 201 06 0 00 007741'                       MOVEI   6,[illegible]S66        ; get sstep table address

                                          ; Loop on single step table entries

007727' 400 15 0 00 000000                TA66:   SETZ    ERFLG,                  ; clear error flag
007730' 260 17 0 00 007564*                       GO      SEXEC                   ; execute table entry
007731' 254 00 0 00 007740'                       JRST    TX66                    ; end of sstep table
007732' 254 00 0 00 007727'                       JRST    TA66                    ; keep looping after call
007733' 474 15 0 00 000000                        SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

007734' 027 00 0 00 007743'                     SCOPER  MA66                    ; print error message
007735' 254 00 0 00 007724'                     JRST    TL66                    ; loop on error
007736' 254 00 0 00 007740'                     JRST    TX66                    ; altmode exit
007737' 322 15 0 00 007727'                     JUMPE   ERFLG,TA66              ; do next sstep table entry

                                        ; End of test

007740' 263 17 0 00 000000              TX66:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

007741' 106201  000000                  TS66:   STABLE  (SSSTRT,^D50,100,0)
007742' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

007743' 140000  011377'                 MA66:   MSG!TXNOT![ASCIZ /Cntr was affected when it shouldn't have been/]
007744' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

007745' 010000  000000                  T66M:   MWORD   <ADDR=100,JZ>                   ; init PC       (100)
007746' 000000  000000
007747' 000000  000000                          MWORD   <ADDR=0,LDCT,J=0>               ;               (0)
007750' 000000  000300
007751' 000100  020000                          MWORD   <JMAP,DISA,CCOF,J=2>            ; disa/off      (1)
007752' 000000  010040
007753' 000200  030000                          MWORD   <JMAP,DISA,CCON,J=3>            ; disa/on       (2)
007754' 000000  030040
007755' 000300  040000                          MWORD   <JMAP,ENA,CCOF,J=4>             ; ena/off       (3)
007756' 000400  010040
007757' 000400  050000                          MWORD   <JMAP,ENA,CCON,J=5>             ; ena/on        (4)
007760' 000400  030040
007761' 000500  060000                          MWORD   <CJS,DISA,CCOF,J=6>             ; disa/off      (5)
007762' 000000  010020
007763' 000600  070000                          MWORD   <CJS,DISA,CCON,J=7>             ; disa/on       (6)
007764' 000000  030020
007765' 000707  770000                          MWORD   <CJS,ENA,CCOF,J=777>            ; ena/off       (7)
007766' 000400  010020
007767' 001000  110000                          MWORD   <CJS,ENA,CCON,J=11>             ; ena/on        (10)
007770' 000400  030020
007771' 001100  120000                          MWORD   <CJP,DISA,CCOF,J=12>            ; disa/off      (11)
007772' 000000  010060
007773' 001200  130000                          MWORD   <CJP,DISA,CCON,J=13>            ; disa/on       (12)
007774' 000000  030060
007775' 001307  770000                          MWORD   <CJP,ENA,CCOF,J=777>            ; ena/off       (13)
007776' 000400  010060
007777' 001400  150000                          MWORD   <CJP,ENA,CCON,J=15>             ; ena/on        (14)
010000' 000400  030060
010001' 001507  770000                          MWORD   <PUSH,ENA,CCOF,J=777>           ; ena/off       (15)
010002' 000400  010100
010003' 001600  170000                          MWORD   <JSRP,DISA,CCOF,J=17>           ; disa/off      (16)
```

```
010004' 000000 010120
010005' 001700 200000          MWORD   <JSRP,DISA,CCON,J=20>          ; disa/on        (17)
010006' 000000 030120
010007' 002000 210000          MWORD   <JSRP,ENA,CCON,J=21>           ; ena/on         (20)
010010' 000400 030120
010011' 002100 220000          MWORD   <CJV,DISA,CCOF,J=22>           ; disa/off       (21)
010012' 000000 010140
010013' 002200 230000          MWORD   <CJV,DISA,CCON,J=23>           ; disa/on        (22)
010014' 000000 030140
010015' 002307 770000          MWORD   <CJV,ENA,CCOF,J=777>           ; ena/off        (23)
010016' 000400 010140
010017' 002400 250000          MWORD   <CJV,ENA,CCON,J=25>            ; ena/on         (24)
010020' 000400 030140
010021' 002500 260000          MWORD   <JRP,DISA,CCOF,J=26>           ; disa/off       (25)
010022' 000000 010160
010023' 002600 270000          MWORD   <JRP,DISA,CCON,J=27>           ; disa/on        (26)
010024' 000000 030160
010025' 002700 300000          MWORD   <JRP,ENA,CCON,J=30>            ; ena/on         (27)
010026' 000400 030160
010027' 003007 770000          MWORD   <RFCT,DISA,CCOF,J=777>         ; disa/off       (30)
010030' 000000 010200
010031' 003107 770000          MWORD   <RFCT,DISA,CCON,J=777>         ; disa/on        (31)
010032' 000000 030200
010033' 003207 770000          MWORD   <RFCT,ENA,CCOF,J=777>          ; ena/off        (32)
010034' 000400 010200
010035' 003307 770000          MWORD   <RFCT,ENA,CCON,J=777>          ; ena/on         (33)
010036' 000400 030200
010037' 003407 770000          MWORD   <RPCT,DISA,CCOF,J=777>         ; disa/off       (34)
010040' 000000 010220
010041' 003507 770000          MWORD   <RPCT,DISA,CCON,J=777>         ; disa/on        (35)
010042' 000000 030220
010043' 003607 770000          MWORD   <RPCT,ENA,CCOF,J=777>          ; ena/off        (36)
010044' 000400 010220
010045' 003707 770000          MWORD   <RPCT,ENA,CCON,J=777>          ; ena/on         (37)
010046' 000400 030220
010047' 004007 770000          MWORD   <CRTN,ENA,CCOF,J=777>          ; ena/off        (40)
010050' 000400 010240
010051' 004100 420000          MWORD   <CJPP,DISA,CCOF,J=42>          ; disa/off       (41)
010052' 000000 010260
010053' 004200 430000          MWORD   <CJPP,DISA,CCON,J=43>          ; disa/on        (42)
010054' 000000 030260
010055' 004307 770000          MWORD   <CJPP,ENA,CCOF,J=777>          ; ena/off        (43)
010056' 000400 010260
010057' 004400 450000          MWORD   <CJPP,ENA,CCON,J=45>           ; ena/on         (44)
010060' 000400 030260
010061' 004507 770000          MWORD   <LOOP,DISA,CCOF,J=777>         ; disa/off       (45)
010062' 000000 010320
010063' 004607 770000          MWORD   <LOOP,DISA,CCON,J=777>         ; disa/on        (46)
010064' 000000 030320
010065' 004707 770000          MWORD   <LOOP,ENA,CCON,J=777>          ; ena/on         (47)
010066' 000400 030320
010067' 005007 770000          MWORD   <CONT,DISA,CCOF,J=777>         ; disa/off       (50)
010070' 000000 010340
010071' 005107 770000          MWORD   <CONT,DISA,CCON,J=777>         ; disa/on        (51)
010072' 000000 030340
```

```
6832  010073' 005207  770000            MWORD   <CONT,ENA,CCOF,J=777>          ; ena/off       (52)
6833  010074' 000400  010340
6834  010075' 005307  770000            MWORD   <CONT,ENA,CCON,J=777>          ; ena/on        (53)
6835  010076' 000400  030340
6836  010077' 005407  770000            MWORD   <TWB,DISA,CCOF,J=777>          ; disa/off      (54)
6837  010100' 000000  010360
6838  010101' 005507  770000            MWORD   <TWB,DISA,CCON,J=777>          ; disa/on       (55)
6839  010102' 000000  030360
6840  010103' 005600  570000            MWORD   <TWB,ENA,CCOF,J=57>            ; ena/off       (56)
6841  010104' 000400  010360
6842  010105' 005707  770000            MWORD   <TWB,ENA,CCON,J=777>           ; ena/on        (57)
6843  010106' 000400  030360
6844
6845  010107' 006007  770000            MWORD   <JRP,ENA,CCOF,J=777>           ; ena/off       (60)
6846  010110' 000400  010160
6847
6848  010111' 077707  770000            MWORD   <ADDR=777,JMAP,J=777>          ; error if here (777)
6849  010112' 000000  000040
6850  010113' 777777  777777            -1
```

```
MA64
MA65
MA66
MA67
MA7
MA70
MA70P1
MA70P2
MB70
MBCN
MLAST

MPROC

MSG

NDMP

ODELAY
P
PARFLG
PORTCI
PORTNI
RDCSR
RDLAR
SAVCSR
SDATA
SEQ

SETLAR
SEXEC

SNEXT
SSCALL
SSCHK
SSCONT

SSJRST
SSLAST
```

```

                                  ;#*********************************************************************
                                  ;* Test 67 - Reg/Ctr Interference Tests
                                  ;
                                  ; Description:  Verify that certain instructions  do  not  affect
                                  ;               the register/counter (initially non-zero).
                                  ;
                                  ; Procedure:    Clear Port
                                  ;               Load test microcode
                                  ;               Set up RAR to start at location 100
                                  ;               Set MPRUN/SINGLE STEP 39 (decimal) times
                                  ;               Read LAR, and verify address = 77
                                  ;
                                  ; Failure:      ---
                                  ;#*********************************************************************

                                  ; Test data

010114' 254 00 0 00 010122'       TSTS67: JRST    TG67                    ; go start test
010115' 240401  000067                    SEQ!MPROC!NDMP!ZSEQ!67          ; test mask
010116' 010150' 011371'                   T67M,,[ASCIZ ^Reg/Ctr Interference Tests^]
010117' 010442' 010517'                   [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
010120' 000000  010273'                   TSTS70                          ; failure test table
010121' 777777  777777                    -1                              ;   ...

                                  ; Start test

010122' 201 00 0 00 000000'       TG67:   MOVEI   Z3                      ; get address of module start
010123' 260 17 0 00 007720*               GO      TRACE                   ; handle trace output
010124' 201 01 0 00 010150'               MOVEI   1,T67M                  ; set up microcode address
010125' 260 17 0 00 007722*               GO      TLOAD                   ; load/verify it
010126' 263 17 0 00 000000                RTN                             ; failed - exit test

                                  ; Initialization

010127' 260 17 0 00 007724*       TL67:   GO      IPACLR                  ; clear port
010130' 402 00 0 00 007725*               SETZM   TSTSUB                  ; initialize subtest number
010131' 201 06 0 00 010144'               MOVEI   6,TS67                  ; get sstep table address

                                  ; Loop on single step table entries

010132' 400 15 0 00 000000        TA67:   SETZ    ERFLG,                  ; clear error flag
010133' 260 17 0 00 007730*               GO      SEXEC                   ; execute table entry
010134' 254 00 0 00 010143'               JRST    TX67                    ; end of sstep table
010135' 254 00 0 00 010132'               JRST    TA67                    ; keep looping after call
010136' 474 15 0 00 000000                SETO    ERFLG,                  ; error - end address incorrect
```

```

                                        ; Handle error printouts and scope looping

010137' 027 00 0 00 010146'                    SCOPER  MA67                    ; print error message
010140' 254 00 0 00 010127'                    JRST    TL67                    ; loop on error
010141' 254 00 0 00 010143'                    JRST    TX67                    ; altmode exit
010142' 322 15 0 00 010132'                    JUMPE   ERFLG,TA67              ; do next sstep table entry

                                        ; End of test

010143' 263 17 0 00 000000              TX67:   RTN                             ; return

                                        ; Single step table, as:  (CMD,#Steps,Start addr,End Addr)

010144' 104601  100077                  TS67:   STABLE  (SSSTRT,^D38,110,77)
010145' 000000  000000                          STABLE  (SSLAST)

                                        ; Error messages

010146' 140000  011377'                 MA67:   MSG!TXNOT![ASCIZ /Cntr was affected when it shouldn't have been/]
010147' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data

                                        ; Test microcode

010150' 011000  000000                  T67M:   MWORD   <ADDR=110,JZ>                   ; init PC       (110)
010151' 000000  000000
010152' 000000  770000                          MWORD   <ADDR=0,LDCT,J=77>              ;               (0)
010153' 000000  000300
010154' 000100  020000                          MWORD   <JMAP,DISA,CCOF,J=2>            ; disa/off      (1)
010155' 000000  010040
010156' 000200  030000                          MWORD   <JMAP,DISA,CCON,J=3>            ; disa/on       (2)
010157' 000000  030040
010160' 000300  040000                          MWORD   <JMAP,ENA,CCOF,J=4>             ; ena/off       (3)
010161' 000400  010040
010162' 000400  050000                          MWORD   <JMAP,ENA,CCON,J=5>             ; ena/on        (4)
010163' 000400  030040
010164' 000500  060000                          MWORD   <CJS,DISA,CCOF,J=6>             ; disa/off      (5)
010165' 000000  010020
010166' 000600  070000                          MWORD   <CJS,DISA,CCON,J=7>             ; disa/on       (6)
010167' 000000  030020
010170' 000707  770000                          MWORD   <CJS,ENA,CCOF,J=777>            ; ena/off       (7)
010171' 000400  010020
010172' 001000  110000                          MWORD   <CJS,ENA,CCON,J=11>             ; ena/on        (10)
010173' 000400  030020
010174' 001100  120000                          MWORD   <CJP,DISA,CCOF,J=12>            ; disa/off      (11)
010175' 000000  010060
010176' 001200  130000                          MWORD   <CJP,DISA,CCON,J=13>            ; disa/on       (12)
010177' 000000  030060
010200' 001307  770000                          MWORD   <CJP,ENA,CCOF,J=777>            ; ena/off       (13)
010201' 000400  010060
010202' 001400  150000                          MWORD   <CJP,ENA,CCON,J=15>             ; ena/on        (14)
010203' 000400  030060
010204' 001507  770000                          MWORD   <PUSH,ENA,CCOF,J=777>           ; disa/on       (15)
010205' 000400  010100
010206' 001600  170000                          MWORD   <JSRP,DISA,CCOF,J=17>           ; disa/off      (16)
```

37

T47M
T4M
T50M
T51M
T52M
T53LOD
T53M
T53MM
T54M
T55M
T56M
T57M
T5M
T60M
T61M
T62M
T63M
T64M
T65M
T66M
T67M
T6M
T70L01
T70LOD
T7M
TA1
TA10
TA11
TA12
TA13
TA14
TA15
TA16
TA17
TA2
TA20
TA21
TA22
TA23
TA24
TA25
TA26
TA3
TA30
TA31
TA32
TA33
TA34
TA35
TA36
TA37
TA4
TA40

838

```
010207' 000000  010120
010210' 001700  200000          MWORD   <JSRP,DISA,CCON,J=20>           ; disa/on       (17)
010211' 000000  030120
010212' 002000  210000          MWORD   <JSRP,ENA,CCON,J=21>            ; ena/on        (20)
010213' 000400  030120
010214' 002100  220000          MWORD   <CJV,DISA,CCOF,J=22>            ; disa/off      (21)
010215' 000000  010140
010216' 002200  230000          MWORD   <CJV,DISA,CCON,J=23>            ; disa/on       (22)
010217' 000000  030140
010220' 002307  770000          MWORD   <CJV,ENA,CCOF,J=777>            ; ena/off       (23)
010221' 000400  010140
010222' 002400  250000          MWORD   <CJV,ENA,CCON,J=25>             ; ena/on        (24)
010223' 000400  030140
010224' 002500  260000          MWORD   <JRP,DISA,CCOF,J=26>            ; disa/off      (25)
010225' 000000  010160
010226' 002600  270000          MWORD   <JRP,DISA,CCON,J=27>            ; disa/on       (26)
010227' 000000  030160
010230' 002700  300000          MWORD   <JRP,ENA,CCON,J=30>             ; ena/on        (27)
010231' 000400  030160
010232' 003007  770000          MWORD   <CRTN,ENA,CCOF,J=777>           ; ena/off       (30)
010233' 000400  010240
010234' 003100  320000          MWORD   <CJPP,DISA,CCOF,J=32>           ; disa/off      (31)
010235' 000000  010260
010236' 003200  330000          MWORD   <CJPP,DISA,CCON,J=33>           ; disa/on       (32)
010237' 000000  030260
010240' 003307  770000          MWORD   <CJPP,ENA,CCOF,J=777>           ; ena/off       (33)
010241' 000400  010260
010242' 003400  350000          MWORD   <CJPP,ENA,CCON,J=35>            ; ena/on        (34)
010243' 000400  030260
010244' 003507  770000          MWORD   <LOOP,DISA,CCOF,J=777>          ; disa/off      (35)
010245' 000000  010320
010246' 003607  770000          MWORD   <LOOP,DISA,CCON,J=777>          ; disa/on       (36)
010247' 000000  030320
010250' 003707  770000          MWORD   <LOOP,ENA,CCON,J=777>           ; ena/on        (37)
010251' 000400  030320
010252' 004007  770000          MWORD   <CONT,DISA,CCOF,J=777>          ; disa/off      (40)
010253' 000000  010340
010254' 004107  770000          MWORD   <CONT,DISA,CCON,J=777>          ; disa/on       (41)
010255' 000000  030340
010256' 004207  770000          MWORD   <CONT,ENA,CCOF,J=777>           ; ena/off       (42)
010257' 000400  010340
010260' 004307  770000          MWORD   <CONT,ENA,CCON,J=777>           ; ena/on        (43)
010261' 000400  030340
010262' 004407  770000          MWORD   <JRP,ENA,CCOF,J=777>            ; disa/off      (44)
010263' 000400  010160

010264' 007701  000000          MWORD   <ADDR=77,JMAP,J=100>            ; done          (77)
010265' 000000  000040
010266' 010001  000000          MWORD   <JMAP,J=100>                    ; error if here (100)
010267' 000000  000040
010270' 077707  770000          MWORD   <ADDR=777,JMAP,J=777>           ; error if here (777)
010271' 000000  000040
010272' 777777  777777          -1
```

TA41
TA42
TA43
TA44
TA45
TA46
TA47
TA5
TA50
TA51
TA52
TA53
TA54
TA55
TA56
TA57
TA6
TA60
TA61
TA62
TA63
TA64
TA65
TA66
TA67
TA7
TA70
TG1
TG10
TG11
TG12
TG13
TG14
TG15
TG16
TG17
TG2
TG20
TG21
TG22
TG23
TG24
TG25
TG26
TG3
TG30
TG31
TG32
TG33
TG34
TG35
TG36
TG37

```
                                          ;#*********************************************************************
                                          ;* TEST 70 - Full Speed Sequencer Test
                                          ;
                                          ; Description:  Load  entire  CRAM with 'CONT,JMAP, and JMAP with
                                          ;               bad parity' sequences.  Then start it up  at full
                                          ;               speed  and let it run for 2  seconds.  Then  stop
                                          ;               and verify that no CRAM PE occurred.
                                          ;
                                          ; Procedure:    Clear Port
                                          ;               Load microcode/set start address 1
                                          ;               Start microcode (will halt with a parity error at
                                          ;                 if the test fails).
                                          ;
                                          ;               The microcode does a CONT
                                          ;                                    JMAP .+2
                                          ;                                    JMAP . (bad parity)
                                          ;                                    ....
                                          ;
                                          ;               All microinstructions that do a jump or  continue
                                          ;               are used in addition to CONT and JMAP.
                                          ;
                                          ; Failure:      ---
                                          ;#*********************************************************************

                                          ; Test data

 010273' 254 00 0 00 010300'              TSTS70: JRST    TG70                          ; go start test
 010274' 240401  000070                           SEQ!MPROC!NDMP!ZSEQ!70                ; test mask
 010275' 000000  011411'                          0,,[ASCIZ ^Full Speed Sequencer Test^]
 010276' 010442' 010517'                          [EXP MLAST!M7],,[EXP M16,M9,M1,MLAST!M8]
 010277' 777777  777777                           -1                                    ; failure test table

                                          ; Start test

 010300' 201 00 0 00 000000'              TG70:   MOVEI   Z3                            ; get address of module start
 010301' 260 17 0 00 010123*                      GO      TRACE                         ; handle trace output

                                          ; Load microcode

 010302' 260 17 0 00 010127*                      GO      IPACLR                        ; issue a port clear
 010303' 260 17 0 00 010366'                      GO      T70LOD                        ; load microcode
```

```

                                        ; 1st segment of test (Segment A) - Start up port

010304' 400 15 0 00 000000              TA70:   SETZ    ERFLG,                  ; clear error flag
010305' 201 00 0 00 000001                      MOVEI   1                       ; set subtest number
010306' 202 00 0 00 010130*                     MOVEM   TSTSUB                  ;   to 1
010307' 260 17 0 00 010302*                     GO      IPACLR                  ; issue a port clear
010310' 201 00 0 00 000001                      MOVEI   1                       ; get a 1
010311' 202 00 0 00 000000*                     MOVEM   SNEXT                   ; set up start location
010312' 201 00 0 00 000010                      MOVEI   10                      ; get a 10
010313' 202 00 0 00 000000*                     MOVEM   SDATA                   ; set up start CSR data
010314' 260 17 0 00 000000*                     GO      IPASRT                  ; start up the port
010315' 474 15 0 00 000000                      SETO    ERFLG,                  ; failed - error reading/writing CSR
010316' 474 15 0 00 000000                      SETO    ERFLG,                  ; failed - port already running
010317' 474 15 0 00 000000                      SETO    ERFLG,                  ; failed - error bits set in CSR
010320' 027 00 0 00 010340'                     SCOPER  MA70                    ; print error message
010321' 254 00 0 00 010304'                     JRST    TA70                    ; loop on error
010322' 254 00 0 00 010337'                     JRST    TX70                    ; altmode exit
010323' 326 15 0 00 010337'                     JUMPN   ERFLG,TX70              ; error already - exit test

                                        ; 1st segment of test (Segment A) - Check if CRAM PE set after 2 seconds

010324' 350 00 0 00 010306*                     AOS     TSTSUB                  ; increment subtest number
010325' 201 00 0 00 003720                      MOVEI   ^D2000                  ; wait 2 seconds
010326' 336 00 0 00 000000*                     SKIPN   UDEBUG                  ; debug mode?
010327' 260 17 0 00 000000*                     GO      ODELAY                  ; no - do the wait
010330' 260 17 0 00 000000*                     GO      RDCSR                   ; read CSR
010331' 474 15 0 00 000000                      SETO    ERFLG,                  ; failed - couldn't read CSR
010332' 603 01 0 00 004000                      TLNE    1,(CRAMPE)              ; got a CRAM PE?
010333' 474 15 0 00 000000                      SETO    ERFLG,                  ; yes - set error flag
010334' 027 00 0 00 010352'                     SCOPER  MB70                    ; print error message
010335' 254 00 0 00 010304'                     JRST    TA70                    ; loop on error
010336' 254 00 0 00 010337'                     JRST    TX70                    ; altmode exit

                                        ; End of test

010337' 263 17 0 00 000000              TX70:   RTN                             ; return

                                        ; Error message - Microcode did not stop start up correctly

010340' 140000  011417'                 MA70:   MSG!TXNOT![ASCIZ /Couldn't successfully start the port/]
010341' 270000  010342'                         LAST!CALL!TXALL!MA70P1

010342' 260 17 0 00 010330*             MA70P1: GO      RDCSR                   ; get CSR contents
010343' 255 00 0 00 000000                      JFCL                            ; error
010344' 202 01 0 00 012215'                     MOVEM   1,SAVCSR#               ; save it
010345' 200 01 0 00 012215'                     MOVE    1,SAVCSR                ; get CSR data
010346' 201 02 0 00 000023                      MOVEI   2,^D19                  ; character count so far
010347' 201 03 0 00 000023                      MOVEI   3,^D19                  ; continuation line column number
010350' 260 17 0 00 000000*                     GO      CSRPNT                  ; go print in English
010351' 263 17 0 00 000000                      RTN                             ; return
```

```
                                        ; Error message - CRAM PE set in CSR register

010352' 140000  011427'                 MB70:   MSG!TXNOT![ASCIZ /2910 missed a jump or did a continue incorrectly/]
010353' 260000  010342'                         CALL!TXALL!MA70P1
010354' 270000  010355'                         LAST!CALL!TXALL!MA70P2

010355' 260 17 0 00 010307*             MA70P2: GO      IPACLR                  ; do a port clear
010356' 260 17 0 00 000000*                     GO      SETLAR                  ; set 'DIAG Sel LAR/SQR'
010357' 255 00 0 00 000000                      JFCL                            ; error accessing CSR
010360' 260 17 0 00 000000*                     GO      RDLAR                   ; read LAR
010361' 037 00 0 00 011441'                     TMSGC   <LAR (location with CRAM PE): >
010362' 242 01 0 00 777777                      LSH     1,-1
010363' 200 00 0 00 000001                      MOVE    1
010364' 037 04 0 00 000000                      PNT4
010365' 263 17 0 00 000000                      RTN

                                        ; Routine to load CRAM locations

010366' 477 06 0 00 000007              T70LOD: SETOB   6,7                     ; initialize CONT,JMAP pointers
010367' 476 00 0 00 000000*                     SETOM   CADDR                   ; initialize CRAM address
010370' 402 00 0 00 000000*                     SETZM   CWORDL                  ; initialize left half
010371' 402 00 0 00 000000*                     SETZM   CWORDR                  ; initialize right half
010372' 350 01 0 00 010367*             T70L01: AOS     1,CADDR                 ; increment CRAM address
010373' 301 01 0 00 007777                      CAIL    1,7777                  ; done yet?
010374' 263 17 0 00 000000                      RTN                             ; yes - return
010375' 261 17 0 00 000000*                     PUT     PARFLG                  ; save state of parity flag
010376' 476 00 0 00 010375*                     SETOM   PARFLG                  ; calculate good parity
010377' 350 00 0 00 000006                      AOS     6                       ; increment CONT pointer
010400' 303 06 0 00 000001                      CAILE   6,1                     ; done with 2 of them?
010401' 400 06 0 00 000000                      SETZ    6,                      ; yes - re-initialize
010402' 350 00 0 00 000007                      AOS     7                       ; increment JMAP pointer
010403' 303 07 0 00 000006                      CAILE   7,6                     ; done with 7 of them?
010404' 400 07 0 00 000000                      SETZ    7,                      ; yes - re-initialize

                                        ; First store the CONT instruction

010405' 271 01 0 00 000002                      ADDI    1,2                     ; point to bad location
010406' 137 01 0 00 011450'                     DPB     1,[POINT 12,CWORDL,17]  ; insert J field
                                                MOVE    [4                      ; get CONT type instruction
010407' 200 00 0 06 011451'                              14](6)
010410' 137 00 0 00 011453'                     DPB     [POINT 4,CWORDR,31]     ; insert CTL field
010411' 260 17 0 00 000000*                     GO      DWCRAM                  ; load it
```

```


                                        ; Now the first JMAP

   010412' 350 01 0 00 010372*                AOS     1,CADDR                 ; increment CRAM address
   010413' 271 01 0 00 000002                 ADDI    1,2                     ; point to next CONT
   010414' 306 01 0 00 007777                 CAIN    1,7777                  ; at last location?
   010415' 400 01 0 00 000000                 SETZ    1,                      ; yes - make jump field point to 0
   010416' 137 01 0 00 011450'                DPB     1,[POINT 12,CWORDL,17]  ; insert J field
                                              MOVE    [1                      ; get JMAP type instruction
                                                        2
                                                        3
                                                        5
                                                        6
                                                        7
   010417' 200 00 0 07 011454'                          11](7)
   010420' 137 00 0 00 011453'                DPB     [POINT 4,CWORDR,31]     ; insert CTL field
   010421' 260 17 0 00 010411*                GO      DWCRAM                  ; load it

                                        ; Now the 2nd JMAP (with bad parity)

   010422' 350 01 0 00 010412*                AOS     1,CADDR                 ; increment CRAM address
   010423' 137 01 0 00 011450'                DPB     1,[POINT 12,CWORDL,17]  ; insert J field
   010424' 201 00 0 00 000002                 MOVEI   2                       ; get JMAP instruction
   010425' 137 00 0 00 011453'                DPB     [POINT 4,CWORDR,31]     ; insert CTL field
   010426' 474 01 0 00 000000                 SETO    1,                      ; set 'force bad parity' flag
   010427' 200 02 0 00 010370*                MOVE    2,CWORDL                ; get left half
   010430' 200 03 0 00 010371*                MOVE    3,CWORDR                ; get right half
   010431' 260 17 0 00 000000*                GO      CALPAR                  ; insert parity
   010432' 402 00 0 00 010376*                SETZM   PARFLG                  ; don't calculate parity
   010433' 260 17 0 00 010421*                GO      DWCRAM                  ; load it
   010434' 262 17 0 00 010432*                GET     PARFLG                  ; restore state of parity flag
   010435' 254 00 0 00 010372'                JRST    T70L01                  ; do next set of locations


                                        ;#***********************************************************************
                                        ; End of MPROC Module Sequencer Tests
                                        ;#***********************************************************************

                                              XLIST

NO ERRORS DETECTED

PROGRAM BREAK IS 012216
CPU TIME USED 05:21.802

171P CORE USED
```

```
DFPTA3   MACRO %53B(1242) 07:54  1-Jul-85 Page S-1                                                        SEQ 0856
DFPTA3   MAC      18-Apr-85 10:58          SYMBOL TABLE

CADDR            010422' ext  MA47             004747'      T11M             001024'      T70LOD           010366'
CALL     200000  000000  spd  MA5              000501'      T12M             001072'      T7M              000701'
CALPAR           010431' ext  MA50             005037'      T13M             001152'      TA1              000017'
CRAMPE   004000  000000  spd  MA51             005423'      T14M             001222'      TA10             000733'
CSRPNT           010350' ext  MA52             005516'      T15M             001326'      TA11             001006'
CWORDL           011450' ext  MA53             005742'      T16M             001520'      TA12             001054'
CWORDR           011453' ext  MA54             006332'      T17M             001712'      TA13             001134'
DATPAT           012214'      MA55             006434'      T1M              000130'      TA14             001204'
DWCRAM           010433' ext  MA56             006744'      T20M             002106'      TA15             001310'
E17              000021  spd  MA57             007034'      T21M             002276'      TA16             001502'
ERFLG            000015       MA6              000624'      T22M             002472'      TA17             001674'
GET      262740  000000       MA60             007106'      T23M             002550'      TA2              000162'
GO       260740  000000       MA61             007164'      T24M             002630'      TA20             002070'
IPACLR           010355' ext  MA62             007260'      T25M             002712'      TA21             002260'
IPASRT           010314' ext  MA63             007360'      T26M             002776'      TA22             002454'
LAST     010000  000000  spd  MA64             007430'      T2M              000203'      TA23             002532'
M1               000031  spd  MA65             007577'      T30M             003071'      TA24             002612'
M16              000050  spd  MA66             007743'      T31M             003161'      TA25             002674'
M7               000037  spd  MA67             010146'      T32M             003275'      TA26             002760'
M8               000040  spd  MA7              000677'      T33M             003353'      TA3              000247'
M9               000041  spd  MA70             010340'      T34M             003417'      TA30             003053'
MA1              000126'      MA70P1           010342'      T35M             003503'      TA31             003143'
MA10             000752'      MA70P2           010355'      T36M             003745'      TA32             003257'
MA11             001022'      MB70             010352'      T37M             004037'      TA33             003335'
MA12             001070'      MBCN             000016       T3M              000270'      TA34             003401'
MA13             001150'      MLAST    400000  000000  spd  T40M             004135'      TA35             003465'
MA14             001220'      MPROC    200000  000000  spd  T41M             004235'      TA36             003727'
MA15             001324'      MSG      100000  000000  spd  T42M             004320'      TA37             004021'
MA16             001516'      NDMP     000400  000000  spd  T43M             004414'      TA4              000350'
MA17             001710'      ODELAY           010327' ext  T44M             004506'      TA40             004117'
MA2              000201'      P                000017       T45M             004567'      TA41             004217'
MA20             002104'      PARFLG           010434' ext  T46M             004655'      TA42             004277'
MA21             002274'      PNT4     037200  000000       T47M             004751'      TA43             004376'
MA22             002470'      PNTMSG   037000  000000       T4M              000371'      TA44             004470'
MA23             002546'      PORTCI           000117' ext  T50M             005041'      TA45             004550'
MA24             002626'      PORTNI           000115' ext  T51M             005425'      TA46             004637'
MA25             002710'      PUT      261740  000000       T52M             005520'      TA47             004733'
MA26             002774'      RDCSR            010342' ext  T53LOD           005720'      TA5              000451'
MA3              000266'      RDLAR            010360' ext  T53M             005744'      TA50             005023'
MA30             003067'      RTN      263740  000000       T53MM            005726'      TA51             005407'
MA31             003157'      SAVCSR           012215'      T54M             006334'      TA52             005477'
MA32             003273'      SCOPER   027000  000000       T55M             006436'      TA53             005675'
MA33             003351'      SDATA            010313' ext  T56M             006746'      TA54             006316'
MA34             003415'      SEQ      040000  000000  spd  T57M             007036'      TA55             006420'
MA35             003501'      SETLAR           010356' ext  T5M              000503'      TA56             006730'
MA36             003743'      SEXEC            010133' ext  T60M             007110'      TA57             007020'
MA37             004035'      SNEXT            010311' ext  T61M             007166'      TA6              000605'
MA4              000367'      SSCALL           000003  spd  T62M             007262'      TA60             007072'
MA40             004133'      SSCHK            000004  spd  T63M             007362'      TA61             007150'
MA41             004233'      SSCONT           000002  spd  T64M             007432'      TA62             007244'
MA42             004316'      SSJRST           000005  spd  T65M             007601'      TA63             007344'
MA43             004412'      SSLAST           000000  spd  T66M             007745'      TA64             007414'
MA44             004504'      SSPNT            000000  ext  T67M             010150'      TA65             007563'
MA45             004565'      SSSTRT           000001  spd  T6M              000626'      TA66             007727'
MA46             004653'      T10M             000754'      T70L01           010372'      TA67             010132'
```

```
TA7         000660'       TG7         000650'       TL7         000655'       TS64        007426'
TA70        010304'       TG70        010300'       TLOAD       010125' ext   TS65        007575'
TG1         000011'       TL1         000013'       TRACE       010301' ext   TS66        007741'
TG10        000723'       TL10        000730'       TS1         000031'       TS67        010144'
TG11        000776'       TL11        001003'       TS10        000745'       TS7         000672'
TG12        001044'       TL12        001051'       TS11        001020'       TSLOD1      000116' ext
TG13        001124'       TL13        001131'       TS12        001066'       TSLOD2      000120' ext
TG14        001174'       TL14        001201'       TS13        001146'       TSTS1       000000' ent
TG15        001300'       TL15        001305'       TS14        001216'       TSTS10      000712' ent
TG16        001472'       TL16        001477'       TS15        001322'       TSTS11      000765' ent
TG17        001664'       TL17        001671'       TS16        001514'       TSTS12      001033' ent
TG2         000152'       TL2         000157'       TS17        001706'       TSTS13      001113' ent
TG20        002060'       TL20        002065'       TS1CHK      000041'       TSTS14      001163' ent
TG21        002250'       TL21        002255'       TS1LOD      000100'       TSTS15      001267' ent
TG22        002444'       TL22        002451'       TS1PAT      000047'       TSTS16      001461' ent
TG23        002522'       TL23        002527'       TS2         000174'       TSTS17      001653' ent
TG24        002602'       TL24        002607'       TS20        002102'       TSTS2       000141' ent
TG25        002664'       TL25        002671'       TS21        002272'       TSTS20      002047' ent
TG26        002750'       TL26        002755'       TS22        002466'       TSTS21      002237' ent
TG3         000237'       TL3         000244'       TS23        002544'       TSTS22      002433' ent
TG30        003043'       TL30        003050'       TS24        002624'       TSTS23      002511' ent
TG31        003133'       TL31        003140'       TS25        002706'       TSTS24      002571' ent
TG32        003247'       TL32        003254'       TS26        002772'       TSTS25      002653' ent
TG33        003325'       TL33        003332'       TS3         000261'       TSTS26      002737' ent
TG34        003371'       TL34        003376'       TS30        003065'       TSTS27      003025' ent
TG35        003455'       TL35        003462'       TS31        003155'       TSTS3       000226' ent
TG36        003717'       TL36        003724'       TS32        003271'       TSTS30      003032' ent
TG37        004011'       TL37        004016'       TS33        003347'       TSTS31      003122' ent
TG4         000340'       TL4         000345'       TS34        003413'       TSTS32      003236' ent
TG40        004107'       TL40        004114'       TS35        003477'       TSTS33      003314' ent
TG41        004207'       TL41        004214'       TS36        003741'       TSTS34      003360' ent
TG42        004267'       TL42        004274'       TS37        004033'       TSTS35      003444' ent
TG43        004366'       TL43        004373'       TS4         000362'       TSTS36      003706' ent
TG44        004460'       TL44        004465'       TS40        004131'       TSTS37      004000' ent
TG45        004540'       TL45        004545'       TS41        004231'       TSTS4       000327' ent
TG46        004627'       TL46        004634'       TS42        004311'       TSTS40      004076' ent
TG47        004723'       TL47        004730'       TS43        004410'       TSTS41      004176' ent
TG5         000441'       TL5         000446'       TS44        004502'       TSTS42      004256' ent
TG50        005013'       TL50        005020'       TS45        004562'       TSTS43      004355' ent
TG51        005377'       TL51        005404'       TS46        004651'       TSTS44      004447' ent
TG52        005467'       TL52        005474'       TS47        004745'       TSTS45      004527' ent
TG53        005664'       TL53        005671'       TS5         000463'       TSTS46      004616' ent
TG54        006306'       TL54        006313'       TS50        005035'       TSTS47      004712' ent
TG55        006410'       TL55        006415'       TS51        005421'       TSTS5       000430' ent
TG56        006720'       TL56        006725'       TS52        005511'       TSTS50      005002' ent
TG57        007010'       TL57        007015'       TS53        005707'       TSTS51      005366' ent
TG6         000575'       TL6         000602'       TS54        006330'       TSTS52      005456' ent
TG60        007062'       TL60        007067'       TS55        006432'       TSTS53      005653' ent
TG61        007140'       TL61        007145'       TS56        006742'       TSTS54      006275' ent
TG62        007234'       TL62        007241'       TS57        007032'       TSTS55      006377' ent
TG63        007334'       TL63        007341'       TS6         000617'       TSTS56      006707' ent
TG64        007404'       TL64        007411'       TS60        007104'       TSTS57      006777' ent
TG65        007553'       TL65        007560'       TS61        007162'       TSTS6       000564' ent
TG66        007717'       TL66        007724'       TS62        007256'       TSTS60      007051' ent
TG67        010122'       TL67        010127'       TS63        007356'       TSTS61      007127' ent
```

```
DFPTA3  MACRO %53B(1242) 07:54  1-Jul-85 Page S-3
DFPTA3  MAC      18-Apr-85 10:58          SYMBOL TABLE

TSTS62          007223' ent  TX61                007161'      .MJ                 007777  spd
TSTS63          007[illegible]23' ent  TX62                007255'      .MJMAP              000017  spd
TSTS64          0073[illegible]3' ent  TX63                007355'      .MJRP               000017  spd
TSTS65          007543' ent  TX64                007425'      .MJSRP              000017  spd
TSTS66          007710' ent  TX65                007574'      .MJZ                000017  spd
TSTS67          010114' ent  TX66                007740'      .MLDCT              000017  spd
TSTS7           000637' ent  TX67                010143'      .MLOOP              000017  spd
TSTS70          010273' ent  TX7                 000671'      .MPUSH              000017  spd
TSTSUB          010324' ext  TX70                010337'      .MRFCT              000017  spd
TX1             000030'      TXALL     060000    000000  spd  .MRPCT              000017  spd
TX10            000744'      TXNOT     040000    000000  spd  .MTWB               000017  spd
TX11            001017'      UDEBUG              010326' ext  .RCCOF              010000  spd
TX12            001065'      Z3                  000000'      .RCCON              030000  spd
TX13            001145'      ZSEQ      000001    000000  spd  .RCENA    000400    000000  spd
TX14            001215'      $ARG2               000777       .RCJP               000060  spd
TX15            001321'      $B                  000052       .RCJPP              000260  spd
TX16            001513'      $CHR                000052       .RCJS               000020  spd
TX17            001705'      $GARG               000001       .RCJV               000140  spd
TX2             000173'      %ADDR               001000  spd  .RCONT              000340  spd
TX20            002101'      %ML       077707    770000  spd  .RCRTN              000240  spd
TX21            002271'      %MR                 000040  spd  .RDISA              000000  spd
TX22            002465'      .LADDR    000100    000000  spd  .RENA     000400    000000  spd
TX23            002543'      .LCCOF              00[illegible]0  spd  .RJ                 000000  spd
TX24            002623'      .LCCON              00[illegible]  spd  .RJMAP              000040  spd
TX25            002705'      .LCENA              0[illegible]0  spd  .RJRP               000160  spd
TX26            002771'      .LCJP               000000  spd  .RJSRP              000120  spd
TX3             000260'      .LCJPP              000000  spd  .RJZ                000000  spd
TX30            003064'      .LCJS               000000  spd  .RLDCT              000300  spd
TX31            003154'      .LCJV               000000  spd  .RLOOP              000320  spd
TX32            003270'      .LCONT              000000  spd  .RPUSH              000100  spd
TX33            003346'      .LCRTN              000000  spd  .RRFCT              000200  spd
TX34            003412'      .LDISA              000000  spd  .RRPCT              000220  spd
TX35            003476'      .LENA               000000  spd  .RTWB               000360  spd
TX36            003740'      .LJ                 010000  spd
TX37            004032'      .LJMAP              000000  spd
TX4             000361'      .LJRP               000000  spd
TX40            004130'      .LJSRP              000000  spd
TX41            004230'      .LJZ                000000  spd
TX42            004310'      .LLDCT              000000  spd
TX43            004407'      .LLOOP              000000  spd
TX44            004501'      .LPUSH              000000  spd
TX45            004561'      .LRFCT              000000  spd
TX46            004650'      .LRPCT              000000  spd
TX47            004744'      .LTWB               000000  spd
TX5             000462'      .MCCOF              000037  spd
TX50            005034'      .MCCON              000037  spd
TX51            005420'      .MCENA              000001  spd
TX52            005510'      .MCJP               000017  spd
TX53            005706'      .MCJPP              000017  spd
TX54            006327'      .MCJS               000017  spd
TX55            006431'      .MCJV               000017  spd
TX56            006741'      .MCONT              000017  spd
TX57            007031'      .MCRTN              000017  spd
TX6             000616'      .MDISA              000001  spd
TX60            007103'      .MENA               000001  spd
```

```
CADDR        21#   7118   7121   7144   7161
CALL        190    294    405    528    661    793    884    975   1063   1145   1236   1323   1435   1600
           1765   1932   2095   2265   2357   2450   2545   2642   2748   2850   2973   3060   3148   3248
           3476   3585   3698   3810   3916   4026   4132   4232   4339   4445   4548   4849   4966   5190
           5517   5628   5876   5976   6062   6156   6261   6369   6451   6598   6742   6917   7088   7102
           7103
CALPAR       21#   7168
CRAMPE     7075
CSRPNT       20#   7096
CWORDL       21#   7119   7136   7148   7162   7166
CWORDR       21#   7120   7139   7156   7164   7167
DATPAT      129#    129    167
DWCRAM       21#   7140   7157   7170
E17          76     243    354    477
ERFL         97     101    108    266    270    277    377    381    388    500    504    511    624    628
            635     765    769    776    856    860    867    947    951    958   1038   1042   1049   1120
           1124    1131   1211   1215   1222   1298   1302   1309   1410   1414   1421   1575   1579   1586
           1740    1744   1751   1907   1911   1918   2070   2074   2081   2240   2244   2251   2332   2336
           2343    2425   2429   2436   2520   2524   2531   2617   2621   2628   2723   2727   2734   2825
           2829    2836   2948   2952   2959   3035   3039   3046   3123   3127   3134   3223   3227   3234
           3451    3455   3462   3560   3564   3571   3673   3677   3684   3785   3789   3796   3888   3892
           3899    4001   4005   4012   4107   4111   4118   4206   4210   4217   4314   4318   4325   4420
           4424    4431   4523   4527   4534   4824   4828   4835   4938   4942   4949   5133   5137   5144
           5492    5496   5503   5603   5607   5614   5851   5855   5862   5951   5955   5962   6037   6041
           6048    6131   6135   6142   6236   6240   6247   6344   6348   6355   6426   6430   6437   6573
           6577    6584   6717   6721   6728   6892   6896   6903   7050   7059   7060   7061   7065   7074
           7076
IPACLR       25#     90    260    371    494    618    759    850    941   1032   1114   1205   1292   1404
           1569    1734   1901   2064   2234   2326   2419   2514   2611   2717   2819   2942   3029   3117
           3217    3445   3554   3667   3779   3882   3995   4101   4200   4308   4414   4517   4818   4932
           5126    5486   5597   5845   5945   6031   6125   6230   6338   6420   6567   6711   6886   7045
           7053    7105
IPASRT       25#   7058
LAST        190     294    405    528    661    793    884    975   1063   1145   1236   1323   1435   1600
           1765    1932   2095   2265   2357   2450   2545   2642   2748   2850   2973   3060   3148   3248
           3476    3585   3698   3810   3916   4026   4132   4232   4339   4445   4548   4849   4966   5190
           5517    5628   5876   5976   6062   6156   6261   6369   6451   6598   6742   6917   7088   7103
M1           76     243    354    477    601    742    833    924   1015   1097   1188   1275   1387   1552
           1717    1884   2047   2217   2309   2402   2497   2594   2700   2802   2925   3012   3100   3200
           3428    3537   3650   3762   3865   3978   4084   4183   4291   4397   4500   4801   4915   5109
           5469    5580   5828   5928   6014   6108   6213   6321   6403   6551   6696   6872   7035
M16          76     243    354    477    601    742    833    924   1015   1097   1188   1275   1387   1552
           1717    1884   2047   2217   2309   2402   2497   2594   2700   2802   2925   3012   3100   3200
           3428    3537   3650   3762   3865   3978   4084   4183   4291   4397   4500   4801   4915   5109
           5469    5580   5828   5928   6014   6108   6213   6321   6403   6551   6696   6872   7035
M7           76     243    354    477    601    742    833    924   1015   1097   1188   1275   1387   1552
           1717    1884   2047   2217   2309   2402   2497   2594   2700   2802   2925   3012   3100   3200
           3428    3537   3650   3762   3865   3978   4084   4183   4291   4397   4500   4801   4915   5109
           5469    5580   5828   5928   6014   6108   6213   6321   6403   6551   6696   6872   7035
M8           76     243    354    477    601    742    833    924   1015   1097   1188   1275   1387   1552
           1717    1884   2047   2217   2309   2402   2497   2594   2700   2802   2925   3012   3100   3200
           3428    3537   3650   3762   3865   3978   4084   4183   4291   4397   4500   4801   4915   5109
           5469    5580   5828   5928   6014   6108   6213   6321   6403   6551   6696   6872   7035
```

```
M9        76    243    354    477    601    742    833    924   1015   1097   1188   1275   1387   1552     SEQ 0860
        1717   1884   2047   2217   2309   2402   2497   2594   2700   2802   2925   3012   3100   3200
        3428   3537   3650   3762   3865   3978   4084   4183   4291   4397   4500   4801   4915   5109
        5469   5580   5828   5928   6014   6108   6213   6321   6403   6551   6696   6872   7035
MA1      105    189#
MA10     955    974#
MA11    1046   1062#
MA12    1128   1144#
MA13    1219   1235#
MA14    1306   1322#
MA15    1418   1434#
MA16    1583   1599#
MA17    1748   1764#
MA2      274    293#
MA20    1915   1931#
MA21    2078   2094#
MA22    2248   2264#
MA23    2340   2356#
MA24    2433   2449#
MA25    2528   2544#
MA26    2625   2641#
MA3      385    404#
MA30    2731   2747#
MA31    2833   2849#
MA32    2956   2972#
MA33    3043   3059#
MA34    3131   3147#
MA35    3231   3247#
MA36    3459   3475#
MA37    3568   3584#
MA4      508    527#
MA40    3681   3697#
MA41    3793   3809#
MA42    3896   3915#
MA43    4009   4025#
MA44    4115   4131#
MA45    4214   4231#
MA46    4322   4338#
MA47    4428   4444#
MA5      632    660#
MA50    4531   4547#
MA51    4832   4848#
MA52    4946   4965#
MA53    5141   5189#
MA54    5500   5516#
MA55    5611   5627#
MA56    5859   5875#
MA57    5959   5975#
MA6      773    792#
MA60    6045   6061#
MA61    6139   6155#
MA62    6244   6260#
MA63    6352   6368#
```

$B

```
MA64       6434   6450#
MA65       6581   6597#
MA66       6725   6741#
MA67       6900   6916#
MA7         864    883#
MA70       7062   7087#
MA70P1     7088   7090#   7102
MA70P2     7103   7105#
MB70       7077   7101#
MBCN        177    179
MLAST        76    243    354    477    601    742    833    924   1015   1097   1188   1275   1387   1552
           1717   1884   2047   2217   2309   2402   2497   2594   2700   2802   2925   3012   3100   3200
           3428   3537   3650   3762   3865   3978   4084   4183   4291   4397   4500   4801   4915   5109
           5469   5580   5828   5928   6014   6108   6213   6321   6403   6551   6696   6872   7035
MPROC        74    241    352    475    599    740    831    922   1013   1095   1186   1273   1385   1550
           1715   1882   2045   2215   2307   2400   2495   2592   2698   2800   2923   3010   3098   3198
           3426   3535   3648   3760   3863   3976   4082   4181   4289   4395   4498   4799   4913   5107
           5467   5578   5826   5926   6012   6106   6211   6319   6401   6549   6694   6870   7033
MSG         189    293    404    527    660    792    883    974   1062   1144   1235   1322   1434   1599
           1764   1931   2094   2264   2356   2449   2544   2641   2747   2849   2972   3059   3147   3247
           3475   3584   3697   3809   3915   4025   4131   4231   4338   4444   4547   4848   4965   5189
           5516   5627   5875   5975   6061   6155   6260   6368   6450   6597   6741   6916   7087   7101
NDMP         74    241    352    475    599    740    831    922   1013   1095   1186   1273   1385   1550
           1715   1882   2045   2215   2307   2400   2495   2592   2698   2800   2923   3010   3098   3198
           3426   3535   3648   3760   3863   3976   4082   4181   4289   4395   4498   4799   4913   5107
           5467   5578   5826   5926   6012   6106   6211   6319   6401   6549   6694   6870   7033
ODELAY      20#   7072
P           131    184
PARFLG      21#   7124   7125   7169   7171
PORTCI      21#    179
PORTNI      21#    177
RDCSR       25#   7073   7090
RDLAR       25#   7108
SAVCSR    7092#   7092   7093
SDATA       25#   7057
SEQ          74    241    352    475    599    740    831    922   1013   1095   1186   1273   1385   1550
           1715   1882   2045   2215   2307   2400   2495   2592   2698   2800   2923   3010   3098   3198
           3426   3535   3648   3760   3863   3976   4082   4181   4289   4395   4498   4799   4913   5107
           5467   5578   5826   5926   6012   6106   6211   6319   6401   6549   6694   6870   7033
SETLAR      25#   7106
SEXEC       20#     98    267    378    501    625    766    857    948   1039   1121   1212   1299   1411
           1576   1741   1908   2071   2241   2333   2426   2521   2618   2724   2826   2949   3036   3124
           3224   3452   3561   3674   3786   3889   4002   4108   4207   4315   4421   4524   4825   4939
           5134   5493   5604   5852   5952   6038   6132   6237   6345   6427   6574   6718   6893
SNEXT       25#   7055
SSCALL     5152   5153   5154   5155   5156   5157   5158   5159
SSCHK       116    117    118
SSCONT      644    645    646    647    785    786    787    788    876    877    878    879    967    968
            969    970
SSJRST      122
SSLAST      123    124    289    290    400    401    523    524    656    657    788    789    879    880
            970    971   1058   1059   1140   1141   1231   1232   1318   1319   1430   1431   1595   1596
           1760   1761   1927   1928   2090   2091   2260   2261   2352   2353   2445   2446   2540   2541
```

```
              2637      2638      2743      2744      2845      2846      2968      2969      3055      3056      3143      3144      3243      3244      SEQ 0862
              3471      3472      3580      3581      3693      3694      3805      3806      3911      3912      4021      4022      4127      4128
              4227      4228      4334      4335      4440      4441      4543      4544      4844      4845      4961      4962      5160      5161
              5512      5513      5623      5624      5871      5872      5971      5972      6057      6058      6151      6152      6256      6257
              6364      6365      6446      6447      6593      6594      6737      6738      6912      6913
SSPNT          20#       190       294       405       528       661       793       884       975      1063      1145      1236      1323      1435
              1600      1765      1932      2095      2265      2357      2450      2545      2642      2748      2850      2973      3060      3148
              3248      3476      3585      3698      3810      3916      4026      4132      4232      4339      4445      4548      4849      4966
              5190      5517      5628      5876      5976      6062      6156      6261      6369      6451      6598      6742      6917
SSSTRT         118       119       120       121       122       285       286       287       288       289       396       397       398       399
               400       519       520       521       522       523       643       644       647       648       649       650       651       652
               653       654       655       656       784       785       875       876       966       967      1057      1058      1139      1140
              1230      1231      1317      1318      1429      1430      1594      1595      1759      1760      1926      1927      2089      2090
              2259      2260      2351      2352      2444      2445      2539      2540      2636      2637      2742      2743      2844      2845
              2967      2968      3054      3055      3142      3143      3242      3243      3470      3471      3579      3580      3692      3693
              3804      3805      3907      3908      3909      3910      3911      4020      4021      4126      4127      4225      4226      4227
              4333      4334      4439      4440      4542      4543      4843      4844      4957      4958      4959      4960      4961      5153
              5154      5155      5156      5157      5158      5159      5160      5511      5512      5622      5623      5870      5871      5970
              5971      6056      6057      6150      6151      6255      6256      6363      6364      6445      6446      6592      6593      6736
              6737      6911      6912
T10M           923       935      979#
T11M          1014      1026     1067#
T12M          1096      1108     1149#
T13M          1187      1199     1240#
T14M          1274      1286     1327#
T15M          1386      1398     1439#
T16M          1551      1563     1604#
T17M          1716      1728     1769#
T1M             75       165       173       181      194#
T20M          1883      1895     1936#
T21M          2046      2058     2099#
T22M          2216      2228     2269#
T23M          2308      2320     2361#
T24M          2401      2413     2454#
T25M          2496      2508     2549#
T26M          2593      2605     2646#
T2M            242       254      298#
T30M          2699      2711     2752#
T31M          2801      2813     2854#
T32M          2924      2936     2977#
T33M          3011      3023     3064#
T34M          3099      3111     3152#
T35M          3199      3211     3252#
T36M          3427      3439     3480#
T37M          3536      3548     3589#
T3M            353       365      409#
T40M          3649      3661     3702#
T41M          3761      3773     3814#
T42M          3864      3876     3920#
T43M          3977      3989     4030#
T44M          4083      4095     4136#
T45M          4182      4194     4238#
T46M          4290      4302     4343#
```

```
T47M       4396    4408    4449#
T4M         476     488     532#
T50M       4499    4511    4552#
T51M       4800    4812    4853#
T52M       4914    4926    4970#
T53LOD     5152    5154    5156     5158     5165#
T53M       5108    5120    5194#
T53MM      5167    5174#
T54M       5468    5480    5521#
T55M       5579    5591    5634#
T56M       5827    5839    5880#
T57M       5927    5939    5980#
T5M         600     612     665#
T60M       6013    6025    6066#
T61M       6107    6119    6160#
T62M       6212    6224    6265#
T63M       6320    6332    6373#
T64M       6402    6414    6455#
T65M       6550    6561    6602#
T66M       6695    6705    6746#
T67M       6871    6880    6921#
T6M         741     753     797#
T70L01     7121#   7172
T70LOD     7046    7117#
T7M         832     844     888#
TA1          97#    100     108
TA10        947#    950     958
TA11       1038#   1041    1049
TA12       1120#   1123    1131
TA13       1211#   1214    1222
TA14       1298#   1301    1309
TA15       1410#   1413    1421
TA16       1575#   1578    1586
TA17       1740#   1743    1751
TA2         266#    269     277
TA20       1907#   1910    1918
TA21       2070#   2073    2081
TA22       2240#   2243    2251
TA23       2332#   2335    2343
TA24       2425#   2428    2436
TA25       2520#   2523    2531
TA26       2617#   2620    2628
TA3         377#    380     388
TA30       2723#   2726    2734
TA31       2825#   2828    2836
TA32       2948#   2951    2959
TA33       3035#   3038    3046
TA34       3123#   3126    3134
TA35       3223#   3226    3234
TA36       3451#   3454    3462
TA37       3560#   3563    3571
TA4         500#    503     511
TA40       3673#   3676    3684
```

TA41	3785#	3788	3796
TA42	3888#	3891	3899
TA43	4001#	4004	4012
TA44	4107#	4110	4118
TA45	4206#	4209	4217
TA46	4314#	4317	4325
TA47	4420#	4423	4431
TA5	624#	627	635
TA50	4523#	4526	4534
TA51	4824#	4827	4835
TA52	4938#	4941	4949
TA53	5133#	5136	5144
TA54	5492#	5495	5503
TA55	5603#	5606	5614
TA56	5851#	5854	5862
TA57	5951#	5954	5962
TA6	765#	768	776
TA60	6037#	6040	6048
TA61	6131#	6134	6142
TA62	6236#	6239	6247
TA63	6344#	6347	6355
TA64	6426#	6429	6437
TA65	6573#	6576	6584
TA66	6717#	6720	6728
TA67	6892#	6895	6903
TA7	856#	859	867
TA70	7050#	7063	7078
TG1	73	85#	
TG10	921	933#	
TG11	1012	1024#	
TG12	1094	1106#	
TG13	1185	1197#	
TG14	1272	1284#	
TG15	1384	1396#	
TG16	1549	1561#	
TG17	1714	1726#	
TG2	240	252#	
TG20	1881	1893#	
TG21	2044	2056#	
TG22	2214	2226#	
TG23	2306	2318#	
TG24	2399	2411#	
TG25	2494	2506#	
TG26	2591	2603#	
TG3	351	363#	
TG30	2697	2709#	
TG31	2799	2811#	
TG32	2922	2934#	
TG33	3009	3021#	
TG34	3097	3109#	
TG35	3197	3209#	
TG36	3425	3437#	
TG37	3534	3546#	

TG4	474	486#
TG40	3647	3659#
TG41	3759	3771#
TG42	3862	3874#
TG43	3975	3987#
TG44	4081	4093#
TG45	4180	4192#
TG46	4288	4300#
TG47	4394	4406#
TG5	598	610#
TG50	4497	4509#
TG51	4798	4810#
TG52	4912	4924#
TG53	5106	5118#
TG54	5466	5478#
TG55	5577	5589#
TG56	5825	5837#
TG57	5925	5937#
TG6	739	751#
TG60	6011	6023#
TG61	6105	6117#
TG62	6210	6222#
TG63	6318	6330#
TG64	6400	6412#
TG65	6548	6559#
TG66	6693	6703#
TG67	6869	6878#
TG7	830	842#
TG70	7032	7040#
TL1	90#	106
TL10	941#	956
TL11	1032#	1047
TL12	1114#	1129
TL13	1205#	1220
TL14	1292#	1307
TL15	1404#	1419
TL16	1569#	1584
TL17	1734#	1749
TL2	260#	275
TL20	1901#	1916
TL21	2064#	2079
TL22	2234#	2249
TL23	2326#	2341
TL24	2419#	2434
TL25	2514#	2529
TL26	2611#	2626
TL3	371#	386
TL30	2717#	2732
TL31	2819#	2834
TL32	2942#	2957
TL33	3029#	3044
TL34	3117#	3132
TL35	3217#	3232

```
TL36       3445#    3460
TL37       3554#    3569
TL4         494#     509
TL40       3667#    3682
TL41       3779#    3794
TL42       3882#    3897
TL43       3995#    4010
TL44       4101#    4116
TL45       4200#    4215
TL46       4308#    4323
TL47       4414#    4429
TL5         618#     633
TL50       4517#    4532
TL51       4818#    4833
TL52       4932#    4947
TL53       5126#    5142
TL54       5486#    5501
TL55       5597#    5612
TL56       5845#    5860
TL57       5945#    5960
TL6         759#     774
TL60       6031#    6046
TL61       6125#    6140
TL62       6230#    6245
TL63       6338#    6353
TL64       6420#    6435
TL65       6567#    6582
TL66       6711#    6726
TL67       6886#    6901
TL7         850#     865
TLOAD        20#     182      255      366      489      613      754      845      936     1027     1109     1200     1287     1399
           1564     1729     1896     2059     2229     2321     2414     2509     2606     2712     2814     2937     3024     3112
           3212     3440     3549     3662     3774     3877     3990     4096     4195     4303     4409     4512     4813     4927
           5121     5168     5481     5592     5840     5940     6026     6120     6225     6333     6415     6562     6706     6881
TRACE        20#      86      253      364      487      611      752      843      934     1025     1107     1198     1285     1397
           1562     1727     1894     2057     2227     2319     2412     2507     2604     2710     2812     2935     3022     3110
           3210     3438     3547     3660     3772     3875     3988     4094     4193     4301     4407     4510     4811     4925
           5119     5479     5590     5838     5938     6024     6118     6223     6331     6413     6560     6704     6879     7041
TS1          92      116#     122      169      170      171      172
TS10        943      966#
TS11       1034     1057#
TS12       1116     1139#
TS13       1207     1230#
TS14       1294     1317#
TS15       1406     1429#
TS16       1571     1594#
TS17       1736     1759#
TS1CHK      116      127#
TS1LOD      117      164#     176
TS1PAT       93      136#
TS2         262      285#
TS20       1903     1926#
TS21       2066     2089#
```

```
TS22       2236    2259#
TS23       2328    2351#
TS24       2421    2444#
TS25       2516    2539#
TS26       2613    2636#
TS3         373     396#
TS30       2719    2742#
TS31       2821    2844#
TS32       2944    2967#
TS33       3031    3054#
TS34       3119    3142#
TS35       3219    3242#
TS36       3447    3470#
TS37       3556    3579#
TS4         496     519#
TS40       3669    3692#
TS41       3781    3804#
TS42       3884    3907#
TS43       3997    4020#
TS44       4103    4126#
TS45       4202    4225#
TS46       4310    4333#
TS47       4416    4439#
TS5         620     643#
TS50       4519    4542#
TS51       4820    4843#
TS52       4934    4957#
TS53       5128    5152#
TS54       5488    5511#
TS55       5599    5622#
TS56       5847    5870#
TS57       5947    5970#
TS6         761     784#
TS60       6033    6056#
TS61       6127    6150#
TS62       6232    6255#
TS63       6340    6363#
TS64       6422    6445#
TS65       6569    6592#
TS66       6713    6736#
TS67       6888    6911#
TS7         852     875#
TSLOD1       20#    178
TSLOD2       20#    180
TSTS1        10      73#
TSTS10       10     481     604     744     834     921#
TSTS11       11     605     745     835     925    1012#
TSTS12       11     746     836     926    1016    1094#
TSTS13       11     837     927    1017    1098    1185#
TSTS14       11     928    1018    1099    1189    1272#
TSTS15       11    1019    1100    1190    1276    1384#
TSTS16       11    1101    1191    1277    1388    1549#
TSTS17       11    1192    1278    1389    1553    1714#
```

```
TSTS2          10        77       240#
TSTS20         11      1279      1390      1554      1718      1881#
TSTS21         12      1391      1555      1719      1885      2044#
TSTS22         12      1556      1720      1886      2048      2214#
TSTS23         12      1721      1887      2049      2218      2306#
TSTS24         12      1888      2050      2219      2310      2399#
TSTS25         12      2051      2220      2311      2403      2494#
TSTS26         12      2221      2312      2404      2498      2591#
TSTS27         12      2674#
TSTS3          10        78       244       351#
TSTS30         12      2313      2405      2499      2595      2697#
TSTS31         13      2406      2500      2596      2701      2799#
TSTS32         13      2501      2597      2702      2803      2922#
TSTS33         13      2598      2703      2804      2926      3009#
TSTS34         13      2704      2805      2927      3013      3097#
TSTS35         13      2806      2928      3014      3101      3197#
TSTS36         13      2929      3015      3102      3201      3425#
TSTS37         13      3016      3103      3202      3429      3534#
TSTS4          10        79       245       355       474#
TSTS40         13      3104      3203      3430      3538      3647#
TSTS41         14      3204      3431      3539      3651      3759#
TSTS42         14      3432      3540      3652      3763      3862#
TSTS43         14      3541      3653      3764      3866      3975#
TSTS44         14      3654      3765      3867      3979      4081#
TSTS45         14      3766      3868      3980      4085      4180#
TSTS46         14      3869      3981      4086      4184      4288#
TSTS47         14      3982      4087      4185      4292      4394#
TSTS5          10        80       246       356       478       598#
TSTS50         14      4088      4186      4293      4398      4497#
TSTS51         15      4187      4294      4399      4501      4798#
TSTS52         15      4295      4400      4502      4802      4912#
TSTS53         15      4401      4503      4803      4916      5106#
TSTS54         15      4504      4804      4917      5110      5466#
TSTS55         15      4805      4918      5111      5470      5577#
TSTS56         15      4919      5112      5471      5581      5825#
TSTS57         15      5113      5472      5582      5829      5925#
TSTS6          10       247       357       479       602       739#
TSTS60         15      5473      5583      5830      5929      6011#
TSTS61         16      5584      5831      5930      6015      6105#
TSTS62         16      5832      5931      6016      6109      6210#
TSTS63         16      5932      6017      6110      6214      6318#
TSTS64         16      6018      6111      6215      6322      6400#
TSTS65         16      6112      6216      6323      6404      6548#
TSTS66         16      6217      6324      6405      6552      6693#
TSTS67         16      6325      6406      6553      6697      6869#
TSTS7          10       358       480       603       743       830#
TSTS70         16      6407      6554      6698      6873      7032#
TSTSUB         20#       91       261       372       495       619       760       851       942      1033      1115      1206      1293      1405
             1570      1735      1902      2065      2235      2327      2420      2515      2612      2718      2820      2943      3030      3118
             3218      3446      3555      3668      3780      3883      3996      4102      4201      4309      4415      4518      4819      4933
             5127      5487      5598      5846      5946      6032      6126      6231      6339      6421      6568      6712      6887      7052
             7069
TX1            99       107       112#
```

```
TX10         949      957      962#
TX11        1040     1048     1053#
TX12        1122     1130     1135#
TX13        1213     1221     1226#
TX14        1300     1308     1313#
TX15        1412     1420     1425#
TX16        1577     1585     1590#
TX17        1742     1750     1755#
TX2          268      276      281#
TX20        1909     1917     1922#
TX21        2072     2080     2085#
TX22        2242     2250     2255#
TX23        2334     2342     2347#
TX24        2427     2435     2440#
TX25        2522     2530     2535#
TX26        2619     2627     2632#
TX3          379      387      392#
TX30        2725     2733     2738#
TX31        2827     2835     2840#
TX32        2950     2958     2963#
TX33        3037     3045     3050#
TX34        3125     3133     3138#
TX35        3225     3233     3238#
TX36        3453     3461     3466#
TX37        3562     3570     3575#
TX4          502      510      515#
TX40        3675     3683     3688#
TX41        3787     3795     3800#
TX42        3890     3898     3903#
TX43        4003     4011     4016#
TX44        4109     4117     4122#
TX45        4208     4216     4221#
TX46        4316     4324     4329#
TX47        4422     4430     4435#
TX5          626      634      639#
TX50        4525     4533     4538#
TX51        4826     4834     4839#
TX52        4940     4948     4953#
TX53        5135     5143     5148#
TX54        5494     5502     5507#
TX55        5605     5613     5618#
TX56        5853     5861     5866#
TX57        5953     5961     5966#
TX6          767      775      780#
TX60        6039     6047     6052#
TX61        6133     6141     6146#
TX62        6238     6246     6251#
TX63        6346     6354     6359#
TX64        6428     6436     6441#
TX65        6575     6583     6588#
TX66        6719     6727     6732#
TX67        6894     6902     6907#
TX7          858      866      871#
```

```
TX70       7064   7065   7079   7083#
TXALL       190    294    405    528    661    793    884    975   1063   1145   1236   1323   1435   1600
           1765   1932   2095   2265   2357   2450   2545   2642   2748   2850   2973   3060   3148   3248
           3476   3585   3698   3810   3916   4026   4132   4232   4339   4445   4548   4849   4966   5190
           5517   5628   5876   5976   6062   6156   6261   6369   6451   6598   6742   6917   7088   7102
           7103
TXNOT       189    293    404    527    660    792    883    974   1062   1144   1235   1322   1434   1599
           1764   1931   2094   2264   2356   2449   2544   2641   2747   2849   2972   3059   3147   3247
           3475   3584   3697   3809   3915   4025   4131   4231   4338   4444   4547   4848   4965   5189
           5516   5627   5875   5975   6061   6155   6260   6368   6450   6597   6741   6916   7087   7101
UDEBUG      21#   7071
Z3          32#     85    252    363    486    610    751    842    933   1024   1106   1197   1284   1396
     [illegible]  1726   1893   2056   2226   2318   2411   2506   2603   2709   2811   2934   3021   3109
     [illegible]  3437   3546   3659   3771   3874   3987   4093   4192   4300   4406   4509   4810   4924
     [illegible]  5478   5589   5837   5937   6023   6117   6222   6330   6412   6559   6703   6878   7040
ZSEQ         74    241    352    475    599    740    831    922   1013   1095   1186   1273   1385   1550
           1715   1882   2045   2215   2307   2400   2495   2592   2675   2698   2800   2923   3010   3098
           3198   3426   3535   3648   3760   3863   3976   4082   4181   4289   4395   4498   4799   4913
           5107   5467   5578   5826   5926   6012   6106   6211   6319   6401   6549   6694   6870   7033
$ARG2       194#   194    196#   196    198#   198    200#   200    298#   298    300#   300    302#   302
            304#   304    306#   306    309#   309    311#   311    313#   313    315#   315    409#   409
            411#   411    413#   413    415#   415    417#   417    419#   419    421#   421    423#   423
            425#   425    427#   427    429#   429    431#   431    433#   433    436#   436    438#   438
            532#   532    534#   534    536#   536    538#   538    540#   540    542#   542    544#   544
            546#   546    548#   548    550#   550    552#   552    554#   554    556#   556    559#   559
            561#   561    665#   665    667#   667    669#   669    671#   671    673#   673    675#   675
            677#   677    679#   679    681#   681    683#   683    685#   685    687#   687    689#   689
            691#   691    693#   693    695#   695    697#   697    699#   699    701#   701    703#   703
            705#   705    707#   707    709#   709    711#   711    797#   797    799#   799    801#   801
            803#   803    888#   888    890#   890    892#   892    894#   894    979#   979    981#   981
            983#   983    985#   985   1067#  1067   1069#  1069   1071#  1071   1149#  1149   1151#  1151
           1153#  1153   1155#  1155   1157#  1157   1159#  1159   1161#  1161   1164#  1164   1240#  1240
           1242#  1242   1244#  1244   1246#  1246   1327#  1327   1329#  1329   1331#  1331   1333#  1333
           1335#  1335   1337#  1337   1339#  1339   1341#  1341   1343#  1343   1345#  1345   1347#  1347
           1349#  1349   1351#  1351   1353#  1353   1356#  1356   1358#  1358   1360#  1360   1363#  1363
           1439#  1439   1441#  1441   1443#  1443   1445#  1445   1447#  1447   1449#  1449   1451#  1451
           1453#  1453   1455#  1455   1457#  1457   1459#  1459   1461#  1461   1463#  1463   1465#  1465
           1467#  1467   1469#  1469   1471#  1471   1473#  1473   1475#  1475   1477#  1477   1479#  1479
           1481#  1481   1483#  1483   1485#  1485   1487#  1487   1489#  1489   1491#  1491   1493#  1493
           1495#  1495   1497#  1497   1499#  1499   1501#  1501   1503#  1503   1505#  1505   1507#  1507
           1509#  1509   1511#  1511   1513#  1513   1515#  1515   1517#  1517   1519#  1519   1521#  1521
           1523#  1523   1526#  1526   1528#  1528   1604#  1604   1606#  1606   1608#  1608   1610#  1610
           1612#  1612   1614#  1614   1616#  1616   1618#  1618   1620#  1620   1622#  1622   1624#  1624
           1626#  1626   1628#  1628   1630#  1630   1632#  1632   1634#  1634   1636#  1636   1638#  1638
           1640#  1640   1642#  1642   1644#  1644   1646#  1646   1648#  1648   1650#  1650   1652#  1652
           1654#  1654   1656#  1656   1658#  1658   1660#  1660   1662#  1662   1664#  1664   1666#  1666
           1668#  1668   1670#  1670   1672#  1672   1674#  1674   1676#  1676   1678#  1678   1680#  1680
           1682#  1682   1684#  1684   1686#  1686   1688#  1688   1691#  1691   1693#  1693   1769#  1769
           1771#  1771   1773#  1773   1775#  1775   1777#  1777   1779#  1779   1781#  1781   1783#  1783
           1785#  1785   1787#  1787   1789#  1789   1791#  1791   1793#  1793   1795#  1795   1797#  1797
           1799#  1799   1801#  1801   1803#  1803   1805#  1805   1807#  1807   1809#  1809   1811#  1811
           1813#  1813   1815#  1815   1817#  1817   1819#  1819   1821#  1821   1823#  1823   1825#  1825
           1827#  1827   1829#  1829   1831#  1831   1833#  1833   1835#  1835   1837#  1837   1839#  1839
```

EQ 0858

SEQ 0871

```
1841#  1841   1843#  1843   1845#  1845   1847#  1847   1849#  1849   1851#  1851   1853#  1853
1855#  1855   1858#  1858   1860#  1860   1936#  1936   1938#  1938   1940#  1940   1942#  1942
1944#  1944   1946#  1946   1948#  1948   1950#  1950   1952#  1952   1954#  1954   1956#  1956
1958#  1958   1960#  1960   1962#  1962   1964#  1964   1966#  1966   1968#  1968   1970#  1970
1972#  1972   1974#  1974   1976#  1976   1978#  1978   1980#  1980   1982#  1982   1984#  1984
1986#  1986   1988#  1988   1990#  1990   1992#  1992   1994#  1994   1996#  1996   1998#  1998
2000#  2000   2002#  2002   2004#  2004   2006#  2006   2008#  2008   2010#  2010   2012#  2012
2014#  2014   2016#  2016   2018#  2018   2021#  2021   2023#  2023   2099#  2099   2101#  2101
2103#  2103   2105#  2105   2107#  2107   2109#  2109   2111#  2111   2113#  2113   2115#  2115
2117#  2117   2119#  2119   2121#  2121   2123#  2123   2125#  2125   2127#  2127   2129#  2129
2131#  2131   2133#  2133   2135#  2135   2137#  2137   2139#  2139   2141#  2141   2143#  2143
2145#  2145   2147#  2147   2149#  2149   2151#  2151   2153#  2153   2155#  2155   2157#  2157
2159#  2159   2161#  2161   2163#  2163   2165#  2165   2167#  2167   2169#  2169   2171#  2171
2173#  2173   2175#  2175   2177#  2177   2179#  2179   2181#  2181   2183#  2183   2185#  2185
2188#  2188   2190#  2190   2269#  2269   2271#  2271   2273#  2273   2275#  2275   2277#  2277
2279#  2279   2281#  2281   2361#  2361   2363#  2363   2365#  2365   2367#  2367   2369#  2369
2371#  2371   2373#  2373   2375#  2375   2454#  2454   2456#  2456   2458#  2458   2460#  2460
2462#  2462   2464#  2464   2466#  2466   2468#  2468   2470#  2470   2549#  2549   2551#  2551
2553#  2553   2555#  2555   2557#  2557   2559#  2559   2561#  2561   2563#  2563   2565#  2565
2567#  2567   2646#  2646   2648#  2648   2650#  2650   2652#  2652   2654#  2654   2656#  2656
2658#  2658   2660#  2660   2662#  2662   2664#  2664   2666#  2666   2752#  2752   2754#  2754
2756#  2756   2758#  2758   2760#  2760   2762#  2762   2764#  2764   2766#  2766   2769#  2769
2771#  2771   2773#  2773   2775#  2775   2854#  2854   2856#  2856   2858#  2858   2860#  2860
2862#  2862   2864#  2864   2866#  2866   2869#  2869   2871#  2871   2873#  2873   2875#  2875
2877#  2877   2879#  2879   2881#  2881   2884#  2834   2886#  2886   2888#  2888   2890#  2890
2892#  2892   2894#  2894   2896#  2896   2899#  2899   2977#  2977   2979#  2979   2981#  2981
2983#  2983   2985#  2985   2987#  2987   2989#  2989   3064#  3064   3066#  3066   3152#  3152
3154#  3154   3156#  3156   3158#  3158   3160#  3160   3162#  3162   3164#  3164   3166#  3166
3168#  3168   3170#  3170   3252#  3252   3254#  3254   3256#  3256   3258#  3258   3260#  3260
3263#  3263   3265#  3265   3267#  3267   3269#  3269   3272#  3272   3274#  3274   3276#  3276
3279#  3279   3281#  3281   3283#  3283   3286#  3286   3288#  3288   3290#  3290   3293#  3293
3295#  3295   3297#  3297   3300#  3300   3302#  3302   3304#  3304   3307#  3307   3309#  3309
3311#  3311   3314#  3314   3316#  3316   3318#  3318   3321#  3321   3323#  3323   3325#  3325
3328#  3328   3330#  3330   3332#  3332   3335#  3335   3337#  3337   3339#  3339   3342#  3342
3344#  3344   3346#  3346   3349#  3349   3351#  3351   3353#  3353   3356#  3356   3358#  3358
3360#  3360   3363#  3363   3365#  3365   3367#  3367   3370#  3370   3372#  3372   3374#  3374
3377#  3377   3379#  3379   3381#  3381   3384#  3384   3386#  3386   3388#  3388   3391#  3391
3393#  3393   3395#  3395   3398#  3398   3400#  3400   3480#  3480   3482#  3482   3484#  3484
3486#  3486   3488#  3488   3490#  3490   3492#  3492   3494#  3494   3496#  3496   3498#  3498
3500#  3500   3503#  3503   3505#  3505   3589#  3589   3591#  3591   3593#  3593   3595#  3595
3598#  3598   3600#  3600   3602#  3602   3605#  3605   3607#  3607   3609#  3609   3612#  3612
3614#  3614   3616#  3616   3619#  3619   3621#  3621   3702#  3702   3705#  3705   3707#  3707
3709#  3709   3712#  3712   3714#  3714   3716#  3716   3719#  3719   3721#  3721   3723#  3723
3726#  3726   3728#  3728   3730#  3730   3733#  3733   3735#  3735   3738#  3738   3814#  3814
3816#  3816   3818#  3818   3820#  3820   3822#  3822   3824#  3824   3826#  3826   3828#  3828
3920#  3920   3922#  3922   3924#  3924   3926#  3926   3928#  3928   3931#  3931   3933#  3933
3935#  3935   3937#  3937   3939#  3939   3941#  3941   3943#  3943   3945#  3945   3948#  3943
4030#  4030   4032#  4032   4034#  4034   4036#  4036   4038#  4038   4040#  4040   4042#  4042
4044#  4044   4046#  4046   4048#  4048   4050#  4050   4052#  4052   4055#  4055   4136#  4136
4138#  4138   4140#  4140   4142#  4142   4144#  4144   4147#  4147   4149#  4149   4151#  4151
4238#  4238   4240#  4240   4242#  4242   4244#  4244   4246#  4246   4251#  4251   4253#  4253
4255#  4255   4257#  4257   4259#  4259   4262#  4262   4343#  4343   4345#  4345   4347#  4347
4349#  4349   4351#  4351   4353#  4353   4355#  4355   4357#  4357   4359#  4359   4361#  4361
```

```
4363#   4363     4366#   4366     4368#   4368     4370#   4370     4449#   4449     4451#   4451     4453#   4453
4455#   4455     4457#   4457     4459#   4459     4461#   4461     4463#   4463     4465#   4465     4467#   4467
4470#   4470     4473#   4473     4552#   4552     4555#   4555     4557#   4557     4559#   4559     4561#   4561
4563#   4563     4565#   4565     4567#   4567     4569#   4569     4572#   4572     4574#   4574     4576#   4576
4578#   4578     4580#   4580     4582#   4582     4584#   4584     4586#   4586     4589#   4589     4591#   4591
4593#   4593     4595#   4595     4597#   4597     4599#   4599     4601#   4601     4603#   4603     4606#   4606
4608#   4608     4610#   4610     4612#   4612     4614#   4614     4616#   4616     4618#   4618     4620#   4620
4623#   4623     4625#   4625     4627#   4627     4629#   4629     4631#   4631     4633#   4633     4635#   4635
4637#   4637     4640#   4640     4642#   4642     4644#   4644     4646#   4646     4648#   4648     4650#   4650
4652#   4652     4654#   4654     4657#   4657     4659#   4659     4661#   4661     4663#   4663     4665#   4665
4667#   4667     4669#   4669     4671#   4671     4674#   4674     4676#   4676     4678#   4678     4680#   4680
4682#   4682     4684#   4684     4686#   4686     4688#   4688     4691#   4691     4693#   4693     4695#   4695
4697#   4697     4699#   4699     4701#   4701     4703#   4703     4705#   4705     4708#   4708     4710#   4710
4712#   4712     4714#   4714     4716#   4716     4718#   4718     4720#   4720     4722#   4722     4725#   4725
4727#   4727     4729#   4729     4731#   4731     4733#   4733     4735#   4735     4737#   4737     4739#   4739
4742#   4742     4744#   4744     4746#   4746     4748#   4748     4750#   4750     4752#   4752     4754#   4754
4756#   4756     4759#   4759     4761#   4761     4763#   4763     4765#   4765     4767#   4767     4769#   4769
4771#   4771     4773#   4773     4775#   4775     4853#   4853     4855#   4855     4857#   4857     4859#   4859
4861#   4861     4863#   4863     4865#   4865     4867#   4867     4869#   4869     4871#   4871     4873#   4873
4875#   4875     4970#   4970     4972#   4972     4974#   4974     4976#   4976     4978#   4978     4980#   4980
4982#   4982     4984#   4984     4986#   4986     4989#   4989     4991#   4991     4993#   4993     4995#   4995
4997#   4997     4999#   4999     5001#   5001     5003#   5003     5005#   5005     5008#   5008     5010#   5010
5012#   5012     5014#   5014     5016#   5016     5018#   5018     5020#   5020     5022#   5022     5024#   5024
5027#   5027     5029#   5029     5031#   5031     5033#   5033     5035#   5035     5037#   5037     5039#   5039
5041#   5041     5043#   5043     5046#   5046     5048#   5048     5050#   5050     5052#   5052     5054#   5054
5056#   5056     5058#   5058     5060#   5060     5063#   5063     5174#   5174     5177#   5177     5180#   5180
5183#   5183     5194#   5194     5196#   5196     5199#   5199     5201#   5201     5203#   5203     5206#   5206
5208#   5208     5210#   5210     5213#   5213     5215#   5215     5217#   5217     5220#   5220     5222#   5222
5224#   5224     5227#   5227     5229#   5229     5231#   5231     5234#   5234     5236#   5236     5238#   5238
5241#   5241     5243#   5243     5245#   5245     5248#   5248     5250#   5250     5252#   5252     5255#   5255
5257#   5257     5259#   5259     5262#   5262     5264#   5264     5266#   5266     5269#   5269     5271#   5271
5273#   5273     5276#   5276     5278#   5278     5280#   5280     5283#   5283     5285#   5285     5287#   5287
5290#   5290     5292#   5292     5294#   5294     5297#   5297     5299#   5299     5301#   5301     5304#   5304
5306#   5306     5308#   5308     5311#   5311     5313#   5313     5315#   5315     5318#   5318     5320#   5320
5322#   5322     5325#   5325     5327#   5327     5329#   5329     5332#   5332     5334#   5334     5336#   5336
5339#   5339     5341#   5341     5343#   5343     5346#   5346     5348#   5348     5350#   5350     5353#   5353
5355#   5355     5357#   5357     5360#   5360     5362#   5362     5364#   5364     5367#   5367     5369#   5369
5371#   5371     5374#   5374     5376#   5376     5378#   5378     5381#   5381     5383#   5383     5386#   5386
5388#   5388     5390#   5390     5392#   5392     5394#   5394     5396#   5396     5398#   5398     5400#   5400
5402#   5402     5404#   5404     5406#   5406     5408#   5408     5410#   5410     5412#   5412     5414#   5414
5416#   5416     5418#   5418     5420#   5420     5422#   5422     5424#   5424     5426#   5426     5428#   5428
5430#   5430     5432#   5432     5434#   5434     5436#   5436     5521#   5521     5523#   5523     5525#   5525
5527#   5527     5529#   5529     5531#   5531     5533#   5533     5535#   5535     5537#   5537     5539#   5539
5542#   5542     5544#   5544     5546#   5546     5548#   5548     5550#   5550     5552#   5552     5554#   5554
5634#   5634     5636#   5636     5638#   5638     5640#   5640     5642#   5642     5644#   5644     5646#   5646
5648#   5648     5650#   5650     5652#   5652     5655#   5655     5657#   5657     5659#   5659     5661#   5661
5663#   5663     5665#   5665     5667#   5667     5669#   5669     5671#   5671     5673#   5673     5675#   5675
5677#   5677     5679#   5679     5681#   5681     5683#   5683     5685#   5685     5687#   5687     5689#   5689
5691#   5691     5693#   5693     5695#   5695     5697#   5697     5699#   5699     5701#   5701     5703#   5703
5705#   5705     5707#   5707     5709#   5709     5711#   5711     5713#   5713     5715#   5715     5717#   5717
5719#   5719     5721#   5721     5723#   5723     5725#   5725     5727#   5727     5729#   5729     5731#   5731
5733#   5733     5735#   5735     5737#   5737     5739#   5739     5741#   5741     5743#   5743     5745#   5745
5747#   5747     5749#   5749     5752#   5752     5754#   5754     5757#   5757     5759#   5759     5761#   5761
```

```
    5763#  5763    5765#  5765    5767#  5767    5769#  5769    5771#  5771    5773#  5773    5775#  5775
    5777#  5777    5779#  5779    5781#  5781    5783#  5783    5785#  5785    5787#  5787    5789#  5789    SEQ 0873
    5791#  5791    5793#  5793    5795#  5795    5797#  5797    5799#  5799    5801#  5801    5803#  5803
    5880#  5880    5882#  5882    5884#  5884    5886#  5886    5888#  5888    5890#  5890    5892#  5892
    5894#  5894    5896#  5896    5898#  5898    5901#  5901    5903#  5903    5980#  5980    5982#  5982
    5984#  5984    5986#  5986    5988#  5988    6066#  6066    6068#  6068    6070#  6070    6072#  6072
    6074#  6074    6076#  6076    6079#  6079    6160#  6160    6162#  6162    6164#  6164    6166#  6166
    6168#  6168    6170#  6170    6172#  6172    6174#  6174    6176#  6176    6179#  6179    6181#  6181
    6183#  6183    6185#  6185    6188#  6188    6265#  6265    6267#  6267    6269#  6269    6271#  6271
    6273#  6273    6275#  6275    6277#  6277    6279#  6279    6281#  6281    6283#  6283    6285#  6285
    6287#  6287    6289#  6289    6291#  6291    6293#  6293    6295#  6295    6373#  6373    6375#  6375
    6377#  6377    6379#  6379    6455#  6455    6457#  6457    6459#  6459    6461#  6461    6463#  6463
    6465#  6465    6467#  6467    6469#  6469    6471#  6471    6473#  6473    6475#  6475    6477#  6477
    6479#  6479    6481#  6481    6483#  6483    6485#  6485    6487#  6487    6489#  6489    6491#  6491
    6493#  6493    6495#  6495    6497#  6497    6499#  6499    6501#  6501    6503#  6503    6505#  6505
    6507#  6507    6509#  6509    6511#  6511    6513#  6513    6515#  6515    6517#  6517    6519#  6519
    6521#  6521    6524#  6524    6527#  6527    6602#  6602    6604#  6604    6606#  6606    6608#  6608
    6610#  6610    6612#  6612    6614#  6614    6616#  6616    6618#  6618    6620#  6620    6622#  6622
    6624#  6624    6626#  6626    6628#  6628    6630#  6630    6632#  6632    6634#  6634    6636#  6636
    6638#  6638    6640#  6640    6642#  6642    6644#  6644    6646#  6646    6648#  6648    6650#  6650
    6652#  6652    6654#  6654    6656#  6656    6658#  6658    6660#  6660    6662#  6662    6664#  6664
    6666#  6666    6669#  6669    6672#  6672    6746#  6746    6748#  6748    6750#  6750    6752#  6752
    6754#  6754    6756#  6756    6758#  6758    6760#  6760    6762#  6762    6764#  6764    6766#  6766
    6768#  6768    6770#  6770    6772#  6772    6774#  6774    6776#  6776    6778#  6778    6780#  6780
    6782#  6782    6784#  6784    6786#  6786    6788#  6788    6790#  6790    6792#  6792    6794#  6794
    6796#  6796    6798#  6798    6800#  6800    6802#  6802    6804#  6804    6806#  6806    6808#  6808
    6810#  6810    6812#  6812    6814#  6814    6816#  6816    6818#  6818    6820#  6820    6822#  6822
    6824#  6824    6826#  6826    6828#  6828    6830#  6830    6832#  6832    6834#  6834    6836#  6836
    6838#  6838    6840#  6840    6842#  6842    6845#  6845    6848#  6848    6921#  6921    6923#  6923
    6925#  6925    6927#  6927    6929#  6929    6931#  6931    6933#  6933    6935#  6935    6937#  6937
    6939#  6939    6941#  6941    6943#  6943    6945#  6945    6947#  6947    6949#  6949    6951#  6951
    6953#  6953    6955#  6955    6957#  6957    6959#  6959    6961#  6961    6963#  6963    6965#  6965
    6967#  6967    6969#  6969    6971#  6971    6973#  6973    6975#  6975    6977#  6977    6979#  6979
    6981#  6981    6983#  6983    6985#  6985    6987#  6987    6989#  6989    6991#  6991    6993#  6993
    6995#  6995    6998#  6998    7000#  7000    7002#  7002
$B   194#   194     196#   196     198#   198     200#   200     298#   298     300#   300     302#   302
     304#   304     306#   306     309#   309     311#   311     313#   313     315#   315     409#   409
     411#   411     413#   413     415#   415     417#   417     419#   419     421#   421     423#   423
     425#   425     427#   427     429#   429     431#   431     433#   433     436#   436     438#   438
     532#   532     534#   534     536#   536     538#   538     540#   540     542#   542     544#   544
     546#   546     548#   548     550#   550     552#   552     554#   554     556#   556     559#   559
     561#   561     665#   665     667#   667     669#   669     671#   671     673#   673     675#   675
     677#   677     679#   679     681#   681     683#   683     685#   685     687#   687     689#   689
     691#   691     693#   693     695#   695     697#   697     699#   699     701#   701     703#   703
     705#   705     707#   707     709#   709     711#   711     797#   797     799#   799     801#   801
     803#   803     888#   888     890#   890     892#   892     894#   894     979#   979     981#   981
     983#   983     985#   985    1067#  1067    1069#  1069    1071#  1071    1149#  1149    1151#  1151
    1153#  1153    1155#  1155    1157#  1157    1159#  1159    1161#  1161    1164#  1164    1240#  1240
    1242#  1242    1244#  1244    1246#  1246    1327#  1327    1329#  1329    1331#  1331    1333#  1333
    1335#  1335    1337#  1337    1339#  1339    1341#  1341    1343#  1343    1345#  1345    1347#  1347
    1349#  1349    1351#  1351    1353#  1353    1356#  1356    1358#  1358    1360#  1360    1363#  1363
    1439#  1439    1441#  1441    1443#  1443    1445#  1445    1447#  1447    1449#  1449    1451#  1451
    1453#  1453    1455#  1455    1457#  1457    1459#  1459    1461#  1461    1463#  1463    1465#  1465
```

%ML

```
1467#  1467   1469#  1469   1471#  1471   1473#  1473   1475#  1475   1477#  1477   1479#  1479
1481#  1481   1483#  1483   1485#  1485   1487#  1487   1489#  1489   1491#  1491   1493#  1493      SEQ 0874
1495#  1495   1497#  1497   1499#  1499   1501#  1501   1503#  1503   1505#  1505   1507#  1507
1509#  1509   1511#  1511   1513#  1513   1515#  1515   1517#  1517   1519#  1519   1521#  1521
1523#  1523   1526#  1526   1528#  1528   1604#  1604   1606#  1606   1608#  1608   1610#  1610
1612#  1612   1614#  1614   1616#  1616   1618#  1618   1620#  1620   1622#  1622   1624#  1624
1626#  1626   1628#  1628   1630#  1630   1632#  1632   1634#  1634   1636#  1636   1638#  1638
1640#  1640   1642#  1642   1644#  1644   1646#  1646   1648#  1648   1650#  1650   1652#  1652
1654#  1654   1656#  1656   1658#  1658   1660#  1660   1662#  1662   1664#  1664   1666#  1666
1668#  1668   1670#  1670   1672#  1672   1674#  1674   1676#  1676   1678#  1678   1680#  1680
1682#  1682   1684#  1684   1686#  1686   1688#  1688   1691#  1691   1693#  1693   1769#  1769
1771#  1771   1773#  1773   1775#  1775   1777#  1777   1779#  1779   1781#  1781   1783#  1783
1785#  1785   1787#  1787   1789#  1789   1791#  1791   1793#  1793   1795#  1795   1797#  1797
1799#  1799   1801#  1801   1803#  1803   1805#  1805   1807#  1807   1809#  1809   1811#  1811
1813#  1813   1815#  1815   1817#  1817   1819#  1819   1821#  1821   1823#  1823   1825#  1825
1827#  1827   1829#  1829   1831#  1831   1833#  1833   1835#  1835   1837#  1837   1839#  1839
1841#  1841   1843#  1843   1845#  1845   1847#  1847   1849#  1849   1851#  1851   1853#  1853
1855#  1855   1858#  1858   1860#  1860   1936#  1936   1938#  1938   1940#  1940   1942#  1942
1944#  1944   1946#  1946   1948#  1948   1950#  1950   1952#  1952   1954#  1954   1956#  1956
1958#  1958   1960#  1960   1962#  1962   1964#  1964   1966#  1966   1968#  1968   1970#  1970
1972#  1972   1974#  1974   1976#  1976   1978#  1978   1980#  1980   1982#  1982   1984#  1984
1986#  1986   1988#  1988   1990#  1990   1992#  1992   1994#  1994   1996#  1996   1998#  1998
2000#  2000   2002#  2002   2004#  2004   2006#  2006   2008#  2008   2010#  2010   2012#  2012
2014#  2014   2016#  2016   2018#  2018   2021#  2021   2023#  2023   2099#  2099   2101#  2101
2103#  2103   2105#  2105   2107#  2107   2109#  2109   2111#  2111   2113#  2113   2115#  2115
2117#  2117   2119#  2119   2121#  2121   2123#  2123   2125#  2125   2127#  2127   2129#  2129
2131#  2131   2133#  2133   2135#  2135   2137#  2137   2139#  2139   2141#  2141   2143#  2143
2145#  2145   2147#  2147   2149#  2149   2151#  2151   2153#  2153   2155#  2155   2157#  2157
2159#  2159   2161#  2161   2163#  2163   2165#  2165   2167#  2167   2169#  2169   2171#  2171
2173#  2173   2175#  2175   2177#  2177   2179#  2179   2181#  2181   2183#  2183   2185#  2185
2188#  2188   2190#  2190   2269#  2269   2271#  2271   2273#  2273   2275#  2275   2277#  2277
2279#  2279   2281#  2281   2361#  2361   2363#  2363   2365#  2365   2367#  2367   2369#  2369
2371#  2371   2373#  2373   2375#  2375   2454#  2454   2456#  2456   2458#  2458   2460#  2460
2462#  2462   2464#  2464   2466#  2466   2468#  2468   2470#  2470   2549#  2549   2551#  2551
2553#  2553   2555#  2555   2557#  2557   2559#  2559   2561#  2561   2563#  2563   2565#  2565
2567#  2567   2646#  2646   2648#  2648   2650#  2650   2652#  2652   2654#  2654   2656#  2656
2658#  2658   2660#  2660   2662#  2662   2664#  2664   2666#  2666   2752#  2752   2754#  2754
2756#  2756   2758#  2758   2760#  2760   2762#  2762   2764#  2764   2766#  2766   2769#  2769
2771#  2771   2773#  2773   2775#  2775   2854#  2854   2856#  2856   2858#  2858   2860#  2860
2862#  2862   2864#  2864   2866#  2866   2869#  2869   2871#  2871   2873#  2873   2875#  2875
2877#  2877   2879#  2879   2881#  2881   2884#  2884   2886#  2886   2888#  2888   2890#  2890
2892#  2892   2894#  2894   2896#  2896   2899#  2899   2977#  2977   2979#  2979   2981#  2981
2983#  2983   2985#  2985   2987#  2987   2989#  2989   3064#  3064   3066#  3066   3152#  3152
3154#  3154   3156#  3156   3158#  3158   3160#  3160   3162#  3162   3164#  3164   3166#  3166
3168#  3168   3170#  3170   3252#  3252   3254#  3254   3256#  3256   3258#  3258   3260#  3260
3263#  3263   3265#  3265   3267#  3267   3269#  3269   3272#  3272   3274#  3274   3276#  3276
3279#  3279   3281#  3281   3283#  3283   3286#  3286   3288#  3288   3290#  3290   3293#  3293
3295#  3295   3297#  3297   3300#  3300   3302#  3302   3304#  3304   3307#  3307   3309#  3309
3311#  3311   3314#  3314   3316#  3316   3318#  3318   3321#  3321   3323#  3323   3325#  3325
3328#  3328   3330#  3330   3332#  3332   3335#  3335   3337#  3337   3339#  3339   3342#  3342
3344#  3344   3346#  3346   3349#  3349   3351#  3351   3353#  3353   3356#  3356   3358#  3358
3360#  3360   3363#  3363   3365#  3365   3367#  3367   3370#  3370   3372#  3372   3374#  3374
3377#  3377   3379#  3379   3381#  3381   3384#  3384   3386#  3386   3388#  3388   3391#  3391
```

```
3393#   3393      3395#   3395      3398#   3398      3400#   3400      3480#   3480      3482#   3482      3484#   3484      SEQ 0875
3486#   3486      3488#   3488      3490#   3490      3492#   3492      3494#   3494      3496#   3496      3498#   3498
3500#   3500      3503#   3503      3505#   3505      3589#   3589      3591#   3591      3593#   3593      3595#   3595
3598#   3598      3600#   3600      3602#   3602      3605#   3605      3607#   3607      3609#   3609      3612#   3612
3614#   3614      3616#   3616      3619#   3619      3621#   3621      3702#   3702      3705#   3705      3707#   3707
3709#   3709      3712#   3712      3714#   3714      3716#   3716      3719#   3719      3721#   3721      3723#   3723
3726#   3726      3728#   3728      3730#   3730      3733#   3733      3735#   3735      3738#   3738      3814#   3814
3816#   3816      3818#   3818      3820#   3820      3822#   3822      3824#   3824      3826#   3826      3828#   3828
3920#   3920      3922#   3922      3924#   3924      3926#   3926      3928#   3928      3931#   3931      3933#   3933
3935#   3935      3937#   3937      3939#   3939      3941#   3941      3943#   3943      3945#   3945      3948#   3948
4030#   4030      4032#   4032      4034#   4034      4036#   4036      4038#   4038      4040#   4040      4042#   4042
4044#   4044      4046#   4046      4048#   4048      4050#   4050      4052#   4052      4055#   4055      4136#   4136
4138#   4138      4140#   4140      4142#   4142      4144#   4144      4147#   4147      4149#   4149      4151#   4151
4238#   4238      4240#   4240      4242#   4242      4244#   4244      4246#   4246      4251#   4251      4253#   4253
4255#   4255      4257#   4257      4259#   4259      4262#   4262      4343#   4343      4345#   4345      4347#   4347
4349#   4349      4351#   4351      4353#   4353      4355#   4355      4357#   4357      4359#   4359      4361#   4361
4363#   4363      4366#   4366      4368#   4368      4370#   4370      4449#   4449      4451#   4451      4453#   4453
4455#   4455      4457#   4457      4459#   4459      5461#   4461      4463#   4463      4465#   4465      4467#   4467
4470#   4470      4473#   4473      4552#   4552      4555#   4555      4557#   4557      4559#   4559      4561#   4561
4563#   4563      4565#   4565      4567#   4567      4569#   4569      4572#   4572      4574#   4574      4576#   4576
4578#   4578      4580#   4580      4582#   4582      4584#   4584      4586#   4586      4589#   4589      4591#   4591
4593#   4593      4595#   4595      4597#   4597      4599#   4599      4601#   4601      4603#   4603      4606#   4606
4608#   4608      4610#   4610      4612#   4612      4614#   4614      4616#   4616      4618#   4618      4620#   4620
4623#   4623      4625#   4625      4627#   4627      4629#   4629      4631#   4631      4633#   4633      4635#   4635
4637#   4637      4640#   4640      4642#   4642      4644#   4644      4646#   4646      4648#   4648      4650#   4650
4652#   4652      4654#   4654      4657#   4657      4659#   4659      4661#   4661      4663#   4663      4665#   4665
4667#   4667      4669#   4669      4671#   4671      4674#   4674      4676#   4676      4678#   4678      4680#   4680
4682#   4682      4684#   4684      4686#   4686      4688#   4688      4691#   4691      4693#   4693      4695#   4695
4697#   4697      4699#   4699      4701#   4701      4703#   4703      4705#   4705      4708#   4708      4710#   4710
4712#   4712      4714#   4714      4716#   4716      4718#   4718      4720#   4720      4722#   4722      4725#   4725
4727#   4727      4729#   4729      4731#   4731      4733#   4733      4735#   4735      4737#   4737      4739#   4739
4742#   4742      4744#   4744      4746#   4746      4748#   4748      4750#   4750      4752#   4752      4754#   4754
4756#   4756      4759#   4759      4761#   4761      4763#   4763      4765#   4765      4767#   4767      4769#   4769
4771#   4771      4773#   4773      4775#   4775      4853#   4853      4855#   4855      4857#   4857      4859#   4859
4861#   4861      4863#   4863      4865#   4865      4867#   4867      4869#   4869      4871#   4871      4873#   4873
4875#   4875      4970#   4970      4972#   4972      4974#   4974      4976#   4976      4978#   4978      4980#   4980
4982#   4982      4984#   4984      4986#   4986      4989#   4989      4991#   4991      4993#   4993      4995#   4995
4997#   4997      4999#   4999      5001#   5001      5003#   5003      5005#   5005      5008#   5008      5010#   5010
5012#   5012      5014#   5014      5016#   5016      5018#   5018      5020#   5020      5022#   5022      5024#   5024
5027#   5027      5029#   5029      5031#   5031      5033#   5033      5035#   5035      5037#   5037      5039#   5039
5041#   5041      5043#   5043      5046#   5046      5048#   5048      5050#   5050      5052#   5052      5054#   5054
5056#   5056      5058#   5058      5060#   5060      5063#   5063      5174#   5174      5177#   5177      5180#   5180
5183#   5183      5194#   5194      5196#   5196      5199#   5199      5201#   5201      5203#   5203      5206#   5206
5208#   5208      5210#   5210      5213#   5213      5215#   5215      5217#   5217      5220#   5220      5222#   5222
5224#   5224      5227#   5227      5229#   5229      5231#   5231      5234#   5234      5236#   5236      5238#   5238
5241#   5241      5243#   5243      5245#   5245      5248#   5248      5250#   5250      5252#   5252      5255#   5255
5257#   5257      5259#   5259      5262#   5262      5264#   5264      5266#   5266      5269#   5269      5271#   5271
5273#   5273      5276#   5276      5278#   5278      5280#   5280      5283#   5283      5285#   5285      5287#   5287
5290#   5290      5292#   5292      5294#   5294      5297#   5297      5299#   5299      5301#   5301      5304#   5304
5306#   5306      5308#   5308      5311#   5311      5313#   5313      5315#   5315      5318#   5318      5320#   5320
5322#   5322      5325#   5325      5327#   5327      5329#   5329      5332#   5332      5334#   5334      5336#   5336
5339#   5339      5341#   5341      5343#   5343      5346#   5346      5348#   5348      5350#   5350      5353#   5353
5355#   5355      5357#   5357      5360#   5360      5362#   5362      5364#   5364      5367#   5367      5369#   5369
```

```
        5371#  5371    5374#  5374    5376#  5376    5378#  5378    5381#  5381    5383#  5383    5386#  5386
        5388#  5388    5390#  5390    5392#  5392    5394#  5394    5396#  5396    5398#  5398    5400#  5400      SEQ 0876
        5402#  5402    5404#  5404    5406#  5406    5408#  5408    5410#  5410    5412#  5412    5414#  5414
        5416#  5416    5418#  5418    5420#  5420    5422#  5422    5424#  5424    5426#  5426    5428#  5428
        5430#  5430    5432#  5432    5434#  5434    5436#  5436    5521#  5521    5523#  5523    5525#  5525
        5527#  5527    5529#  5529    5531#  5531    5533#  5533    5535#  5535    5537#  5537    5539#  5539
        5542#  5542    5544#  5544    5546#  5546    5548#  5548    5550#  5550    5552#  5552    5554#  5554
        5634#  5634    5636#  5636    5638#  5638    5640#  5640    5642#  5642    5644#  5644    5646#  5646
        5648#  5648    5650#  5650    5652#  5652    5655#  5655    5657#  5657    5659#  5659    5661#  5661
        5663#  5663    5665#  5665    5667#  5667    5669#  5669    5671#  5671    5673#  5673    5675#  5675
        5677#  5677    5679#  5679    5681#  5681    5683#  5683    5685#  5685    5687#  5687    5689#  5689
        5691#  5691    5693#  5693    5695#  5695    5697#  5697    5699#  5699    5701#  5701    5703#  5703
        5705#  5705    5707#  5707    5709#  5709    5711#  5711    5713#  5713    5715#  5715    5717#  5717
        5719#  5719    5721#  5721    5723#  5723    5725#  5725    5727#  5727    5729#  5729    5731#  5731
        5733#  5733    5735#  5735    5737#  5737    5739#  5739    5741#  5741    5743#  5743    5745#  5745
        5747#  5747    5749#  5749    5752#  5752    5754#  5754    5757#  5757    5759#  5759    5761#  5761
        5763#  5763    5765#  5765    5767#  5767    5769#  5769    5771#  5771    5773#  5773    5775#  5775
        5777#  5777    5779#  5779    5781#  5781    5783#  5783    5785#  5785    5787#  5787    5789#  5789
        5791#  5791    5793#  5793    5795#  5795    5797#  5797    5799#  5799    5801#  5801    5803#  5803
        5880#  5880    5882#  5882    5884#  5884    5886#  5886    5888#  5888    5890#  5890    5892#  5892
        5894#  5894    5896#  5896    5898#  5898    5901#  5901    5903#  5903    5980#  5980    5982#  5982
        5984#  5984    5986#  5986    5988#  5988    6066#  6066    6068#  6068    6070#  6070    6072#  6072
        6074#  6074    6076#  6076    6079#  6079    6160#  6160    6162#  6162    6164#  6164    6166#  6166
        6168#  6168    6170#  6170    6172#  6172    6174#  6174    6176#  6176    6179#  6179    6181#  6181
        6183#  6183    6185#  6185    6188#  6188    6265#  6265    6267#  6267    6269#  6269    6271#  6271
        6273#  6273    6275#  6275    6277#  6277    6279#  6279    6281#  6281    6283#  6283    6285#  6285
        6287#  6287    6289#  6289    6291#  6291    6293#  6293    6295#  6295    6373#  6373    6375#  6375
        6377#  6377    6379#  6379    6455#  6455    6457#  6457    6459#  6459    6461#  6461    6463#  6463
        6465#  6465    6467#  6467    6469#  6469    6471#  6471    6473#  6473    6475#  6475    6477#  6477
        6479#  6479    6481#  6481    6483#  6483    6485#  6485    6487#  6487    6489#  6489    6491#  6491
        6493#  6493    6495#  6495    6497#  6497    6499#  6499    6501#  6501    6503#  6503    6505#  6505
        6507#  6507    6509#  6509    6511#  6511    6513#  6513    6515#  6515    6517#  6517    6519#  6519
        6521#  6521    6524#  6524    6527#  6527    6602#  6602    6604#  6604    6606#  6606    6608#  6608
        6610#  6610    6612#  6612    6614#  6614    6616#  6616    6618#  6618    6620#  6620    6622#  6622
        6624#  6624    6626#  6626    6628#  6628    6630#  6630    6632#  6632    6634#  6634    6636#  6636
        6638#  6638    6640#  6640    6642#  6642    6644#  6644    6646#  6646    6648#  6648    6650#  6650
        6652#  6652    6654#  6654    6656#  6656    6658#  6658    6660#  6660    6662#  6662    6664#  6664
        6666#  6666    6669#  6669    6672#  6672    6746#  6746    6748#  6748    6750#  6750    6752#  6752
        6754#  6754    6756#  6756    6758#  6758    6760#  6760    6762#  6762    6764#  6764    6766#  6766
        6768#  6768    6770#  6770    6772#  6772    6774#  6774    6776#  6776    6778#  6778    6780#  6780
        6782#  6782    6784#  6784    6786#  6786    6788#  6788    6790#  6790    6792#  6792    6794#  6794
        6796#  6796    6798#  6798    6800#  6800    6802#  6802    6804#  6804    6806#  6806    6808#  6808
        6810#  6810    6812#  6812    6814#  6814    6816#  6816    6818#  6818    6820#  6820    6822#  6822
        6824#  6824    6826#  6826    6828#  6828    6830#  6830    6832#  6832    6834#  6834    6836#  6836
        6838#  6838    6840#  6840    6842#  6842    6845#  6845    6848#  6848    6921#  6921    6923#  6923
        6925#  6925    6927#  6927    6929#  6929    6931#  6931    6933#  6933    6935#  6935    6937#  6937
        6939#  6939    6941#  6941    6943#  6943    6945#  6945    6947#  6947    6949#  6949    6951#  6951
        6953#  6953    6955#  6955    6957#  6957    6959#  6959    6961#  6961    6963#  6963    6965#  6965
        6967#  6967    6969#  6969    6971#  6971    6973#  6973    6975#  6975    6977#  6977    6979#  6979
        6981#  6981    6983#  6983    6985#  6985    6987#  6987    6989#  6989    6991#  6991    6993#  6993
        6995#  6995    6998#  6998    7000#  7000    7002#  7002
$CHR     194#   194     196#   196     198#   198     200#   200     298#   298     300#   300     302#   302
         304#   304     306#   306     309#   309     311#   311     313#   313     315#   315     409#   409
```

```
 411#    411     413#    413     415#    415     417#    417     419#    419     421#    421     423#    423     SEQ 0877
 425#    425     427#    427     429#    429     431#    431     433#    433     436#    436     438#    438
 532#    532     534#    534     536#    536     538#    538     540#    540     542#    542     544#    544
 546#    546     548#    548     550#    550     552#    552     554#    554     556#    556     559#    559
 561#    561     665#    665     667#    667     669#    669     671#    671     673#    673     675#    675
 677#    677     679#    679     681#    681     683#    683     685#    685     687#    687     689#    689
 691#    691     693#    693     695#    695     697#    697     699#    699     701#    701     703#    703
 705#    705     707#    707     709#    709     711#    711     797#    797     799#    799     801#    801
 803#    803     888#    888     890#    890     892#    892     894#    894     979#    979     981#    981
 983#    983     985#    985    1067#   1067    1069#   1069    1071#   1071    1149#   1149    1151#   1151
1153#   1153    1155#   1155    1157#   1157    1159#   1159    1161#   1161    1164#   1164    1240#   1240
1242#   1242    1244#   1244    1246#   1246    1327#   1327    1329#   1329    1331#   1331    1333#   1333
1335#   1335    1337#   1337    1339#   1339    1341#   1341    1343#   1343    1345#   1345    1347#   1347
1349#   1349    1351#   1351    1353#   1353    1356#   1356    1358#   1358    1360#   1360    1363#   1363
1439#   1439    1441#   1441    1443#   1443    1445#   1445    1447#   1447    1449#   1449    1451#   1451
1453#   1453    1455#   1455    1457#   1457    1459#   1459    1461#   1461    1463#   1463    1465#   1465
1467#   1467    1469#   1469    1471#   1471    1473#   1473    1475#   1475    1477#   1477    1479#   1479
1481#   1481    1483#   1483    1485#   1485    1487#   1487    1489#   1489    1491#   1491    1493#   1493
1495#   1495    1497#   1497    1499#   1499    1501#   1501    1503#   1503    1505#   1505    1507#   1507
1509#   1509    1511#   1511    1513#   1513    1515#   1515    1517#   1517    1519#   1519    1521#   1521
1523#   1523    1526#   1526    1528#   1528    1604#   1604    1606#   1606    1608#   1608    1610#   1610
1612#   1612    1614#   1614    1616#   1616    1618#   1618    1620#   1620    1622#   1622    1624#   1624
1626#   1626    1628#   1628    1630#   1630    1632#   1632    1634#   1634    1636#   1636    1638#   1638
1640#   1640    1642#   1642    1644#   1644    1646#   1646    1648#   1648    1650#   1650    1652#   1652
1654#   1654    1656#   1656    1658#   1658    1660#   1660    1662#   1662    1664#   1664    1666#   1666
1668#   1668    1670#   1670    1672#   1672    1674#   1674    1676#   1676    1678#   1678    1680#   1680
1682#   1682    1684#   1684    1686#   1686    1688#   1688    1691#   1691    1693#   1693    1769#   1769
1771#   1771    1773#   1773    1775#   1775    1777#   1777    1779#   1779    1781#   1781    1783#   1783
1785#   1785    1787#   1787    1789#   1789    1791#   1791    1793#   1793    1795#   1795    1797#   1797
1799#   1799    1801#   1801    1803#   1803    1805#   1805    1807#   1807    1809#   1809    1811#   1811
1813#   1813    1815#   1815    1817#   1817    1819#   1819    1821#   1821    1823#   1823    1825#   1825
1827#   1827    1829#   1829    1831#   1831    1833#   1833    1835#   1835    1837#   1837    1839#   1839
1841#   1841    1843#   1843    1845#   1845    1847#   1847    1849#   1849    1851#   1851    1853#   1853
1855#   1855    1858#   1858    1860#   1860    1936#   1936    1938#   1938    1940#   1940    1942#   1942
1944#   1944    1946#   1946    1948#   1948    1950#   1950    1952#   1952    1954#   1954    1956#   1956
1958#   1958    1960#   1960    1962#   1962    1964#   1964    1966#   1966    1968#   1968    1970#   1970
1972#   1972    1974#   1974    1976#   1976    1978#   1978    1980#   1980    1982#   1982    1984#   1984
1986#   1986    1988#   1988    1990#   1990    1992#   1992    1994#   1994    1996#   1996    1998#   1998
2000#   2000    2002#   2002    2004#   2004    2006#   2006    2008#   2008    2010#   2010    2012#   2012
2014#   2014    2016#   2016    2018#   2018    2021#   2021    2023#   2023    2099#   2099    2101#   2101
2103#   2103    2105#   2105    2107#   2107    2109#   2109    2111#   2111    2113#   2113    2115#   2115
2117#   2117    2119#   2119    2121#   2121    2123#   2123    2125#   2125    2127#   2127    2129#   2129
2131#   2131    2133#   2133    2135#   2135    2137#   2137    2139#   2139    2141#   2141    2143#   2143
2145#   2145    2147#   2147    2149#   2149    2151#   2151    2153#   2153    2155#   2155    2157#   2157
2159#   2159    2161#   2161    2163#   2163    2165#   2165    2167#   2167    2169#   2169    2171#   2171
2173#   2173    2175#   2175    2177#   2177    2179#   2179    2181#   2181    2183#   2183    2185#   2185
2188#   2188    2190#   2190    2269#   2269    2271#   2271    2273#   2273    2275#   2275    2277#   2277
2279#   2279    2281#   2281    2361#   2361    2363#   2363    2365#   2365    2367#   2367    2369#   2369
2371#   2371    2373#   2373    2375#   2375    2454#   2454    2456#   2456    2458#   2458    2460#   2460
2462#   2462    2464#   2464    2466#   2466    2468#   2468    2470#   2470    2549#   2549    2551#   2551
2553#   2553    2555#   2555    2557#   2557    2559#   2559    2561#   2561    2563#   2563    2565#   2565
2567#   2567    2646#   2646    2648#   2648    2650#   2650    2652#   2652    2654#   2654    2656#   2656
2658#   2658    2660#   2660    2662#   2662    2664#   2664    2666#   2666    2752#   2752    2754#   2754
```

```
2756#   2756    2758#   2758    2760#   2760    2762#   2762    2764#   2764    2766#   2766    2769#   2769      SEQ 0878
2771#   2771    2773#   2773    2775#   2775    2854#   2854    2856#   2856    2858#   2858    2860#   2860
2862#   2862    2864#   2864    2866#   2866    2869#   2869    2871#   2871    2873#   2873    2875#   2875
2877#   2877    2879#   2879    2881#   2881    2884#   2884    2886#   2886    2888#   2888    2890#   2890
2892#   2892    2894#   2894    2896#   2896    2899#   2899    2977#   2977    2979#   2979    2981#   2981
2983#   2983    2985#   2985    2987#   2987    2989#   2989    3064#   3064    3066#   3066    3152#   3152
3154#   3154    3156#   3156    3158#   3158    3160#   3160    3162#   3162    3164#   3164    3166#   3166
3168#   3168    3170#   3170    3252#   3252    3254#   3254    3256#   3256    3258#   3258    3260#   3260
3263#   3263    3265#   3265    3267#   3267    3269#   3269    3272#   3272    3274#   3274    3276#   3276
3279#   3279    3281#   3281    3283#   3283    3286#   3286    3288#   3288    3290#   3290    3293#   3293
3295#   3295    3297#   3297    3300#   3300    3302#   3302    3304#   3304    3307#   3307    3309#   3309
3311#   3311    3314#   3314    3316#   3316    3318#   3318    3321#   3321    3323#   3323    3325#   3325
3328#   3328    3330#   3330    3332#   3332    3335#   3335    3337#   3337    3339#   3339    3342#   3342
3344#   3344    3346#   3346    3349#   3349    3351#   3351    3353#   3353    3356#   3356    3358#   3358
3360#   3360    3363#   3363    3365#   3365    3367#   3367    3370#   3370    3372#   3372    3374#   3374
3377#   3377    3379#   3379    3381#   3381    3384#   3384    3386#   3386    3388#   3388    3391#   3391
3393#   3393    3395#   3395    3398#   3398    3400#   3400    3480#   3480    3482#   3482    3484#   3484
3486#   3486    3488#   3488    3490#   3490    3492#   3492    3494#   3494    3496#   3496    3498#   3498
3500#   3500    3503#   3503    3505#   3505    3589#   3589    3591#   3591    3593#   3593    3595#   3595
3598#   3598    3600#   3600    3602#   3602    3605#   3605    3607#   3607    3609#   3609    3612#   3612
3614#   3614    3616#   3616    3619#   3619    3621#   3621    3702#   3702    3705#   3705    3707#   3707
3709#   3709    3712#   3712    3714#   3714    3716#   3716    3719#   3719    3721#   3721    3723#   3723
3726#   3726    3728#   3728    3730#   3730    3733#   3733    3735#   3735    3738#   3738    3814#   3814
3816#   3816    3818#   3818    3820#   3820    3822#   3822    3824#   3824    3826#   3826    3828#   3828
3920#   3920    3922#   3922    3924#   3924    3926#   3926    3928#   3928    3931#   3931    3933#   3933
3935#   3935    3937#   3937    3939#   3939    3941#   3941    3943#   3943    3945#   3945    3948#   3948
4030#   4030    4032#   4032    4034#   4034    4036#   4036    4038#   4038    4040#   4040    4042#   4042
4044#   4044    4046#   4046    4048#   4048    4050#   4050    4052#   4052    4055#   4055    4136#   4136
4138#   4138    4140#   4140    4142#   4142    4144#   4144    4147#   4147    4149#   4149    4151#   4151
4238#   4238    4240#   4240    4242#   4242    4244#   4244    4246#   4246    4251#   4251    4253#   4253
4255#   4255    4257#   4257    4259#   4259    4262#   4262    4343#   4343    4345#   4345    4347#   4347
4349#   4349    4351#   4351    4353#   4353    4355#   4355    4357#   4357    4359#   4359    4361#   4361
4363#   4363    4366#   4366    4368#   4368    4370#   4370    4449#   4449    4451#   4451    4453#   4453
4455#   4455    4457#   4457    4459#   4459    4461#   4461    4463#   4463    4465#   4465    4467#   4467
4470#   4470    4473#   4473    4552#   4552    4555#   4555    4557#   4557    4559#   4559    4561#   4561
4563#   4563    4565#   4565    4567#   4567    4569#   4569    4572#   4572    4574#   4574    4576#   4576
4578#   4578    4580#   4580    4582#   4582    4584#   4584    4586#   4586    4589#   4589    4591#   4591
4593#   4593    4595#   4595    4597#   4597    4599#   4599    4601#   4601    4603#   4603    4606#   4606
4608#   4608    4610#   4610    4612#   4612    4614#   4614    4616#   4616    4618#   4618    4620#   4620
4623#   4623    4625#   4625    4627#   4627    4629#   4629    4631#   4631    4633#   4633    4635#   4635
4637#   4637    4640#   4640    4642#   4642    4644#   4644    4646#   4646    4648#   4648    4650#   4650
4652#   4652    4654#   4654    4657#   4657    4659#   4659    4661#   4661    4663#   4663    4665#   4665
4667#   4667    4669#   4669    4671#   4671    4674#   4674    4676#   4676    4678#   4678    4680#   4680
4682#   4682    4684#   4684    4686#   4686    4688#   4688    4691#   4691    4693#   4693    4695#   4695
4697#   4697    4699#   4699    4701#   4701    4703#   4703    4705#   4705    4708#   4708    4710#   4710
4712#   4712    4714#   4714    4716#   4716    4718#   4718    4720#   4720    4722#   4722    4725#   4725
4727#   4727    4729#   4729    4731#   4731    4733#   4733    4735#   4735    4737#   4737    4739#   4739
4742#   4742    4744#   4744    4746#   4746    4748#   4748    4750#   4750    4752#   4752    4754#   4754
4756#   4756    4759#   4759    4761#   4761    4763#   4763    4765#   4765    4767#   4767    4769#   4769
4771#   4771    4773#   4773    4775#   4775    4853#   4853    4855#   4855    4857#   4857    4859#   4859
4861#   4861    4863#   4863    4865#   4865    4867#   4867    4869#   4869    4871#   4871    4873#   4873
4875#   4875    4970#   4970    4972#   4972    4974#   4974    4976#   4976    4978#   4978    4980#   4980
4982#   4982    4984#   4984    4986#   4986    4989#   4989    4991#   4991    4993#   4993    4995#   4995
```

```
0866
4997#  4997   4999#  4999   5001#  5001   5003#  5003   5005#  5005   5008#  5008   5010#  5010     SEQ 0879
5012#  5012   5014#  5014   5016#  5016   5018#  5018   5020#  5020   5022#  5022   5024#  5024
5027#  5027   5029#  5029   5031#  5031   5033#  5033   5035#  5035   5037#  5037   5039#  5039
5041#  5041   5043#  5043   5046#  5046   5048#  5048   5050#  5050   5052#  5052   5054#  5054
5056#  5056   5058#  5058   5060#  5060   5063#  5063   5174#  5174   5177#  5177   5180#  5180
5183#  5183   5194#  5194   5196#  5196   5199#  5199   5201#  5201   5203#  5203   5206#  5206
5208#  5208   5210#  5210   5213#  5213   5215#  5215   5217#  5217   5220#  5220   5222#  5222
5224#  5224   5227#  5227   5229#  5229   5231#  5231   5234#  5234   5236#  5236   5238#  5238
5241#  5241   5243#  5243   5245#  5245   5248#  5248   5250#  5250   5252#  5252   5255#  5255
5257#  5257   5259#  5259   5262#  5262   5264#  5264   5266#  5266   5269#  5269   5271#  5271
5273#  5273   5276#  5276   5278#  5278   5280#  5280   5283#  5283   5285#  5285   5287#  5287
5290#  5290   5292#  5292   5294#  5294   5297#  5297   5299#  5299   5301#  5301   5304#  5304
5306#  5306   5308#  5308   5311#  5311   5313#  5313   5315#  5315   5318#  5318   5320#  5320
5322#  5322   5325#  5325   5327#  5327   5329#  5329   5332#  5332   5334#  5334   5336#  5336
5339#  5339   5341#  5341   5343#  5343   5346#  5346   5348#  5348   5350#  5350   5353#  5353
5355#  5355   5357#  5357   5360#  5360   5362#  5362   5364#  5364   5367#  5367   5369#  5369
5371#  5371   5374#  5374   5376#  5376   5378#  5378   5381#  5381   5383#  5383   5386#  5386
5388#  5388   5390#  5390   5392#  5392   5394#  5394   5396#  5396   5398#  5398   5400#  5400
5402#  5402   5404#  5404   5406#  5406   5408#  5408   5410#  5410   5412#  5412   5414#  5414
5416#  5416   5418#  5418   5420#  5420   5422#  5422   5424#  5424   5426#  5426   5428#  5428
5430#  5430   5432#  5432   5434#  5434   5436#  5436   5521#  5521   5523#  5523   5525#  5525
5527#  5527   5529#  5529   5531#  5531   5533#  5533   5535#  5535   5537#  5537   5539#  5539
5542#  5542   5544#  5544   5546#  5546   5548#  5548   5550#  5550   5552#  5552   5554#  5554
5634#  5634   5636#  5636   5638#  5638   5640#  5640   5642#  5642   5644#  5644   5646#  5646
5648#  5648   5650#  5650   5652#  5652   5655#  5655   5657#  5657   5659#  5659   5661#  5661
5663#  5663   5665#  5665   5667#  5667   5669#  5669   5671#  5671   5673#  5673   5675#  5675
5677#  5677   5679#  5679   5681#  5681   5683#  5683   5685#  5685   5687#  5687   5689#  5689
5691#  5691   5693#  5693   5695#  5695   5697#  5697   5699#  5699   5701#  5701   5703#  5703
5705#  5705   5707#  5707   5709#  5709   5711#  5711   5713#  5713   5715#  5715   5717#  5717
5719#  5719   5721#  5721   5723#  5723   5725#  5725   5727#  5727   5729#  5729   5731#  5731
5733#  5733   5735#  5735   5737#  5737   5739#  5739   5741#  5741   5743#  5743   5745#  5745
5747#  5747   5749#  5749   5752#  5752   5754#  5754   5757#  5757   5759#  5759   5761#  5761
5763#  5763   5765#  5765   5767#  5767   5769#  5769   5771#  5771   5773#  5773   5775#  5775
5777#  5777   5779#  5779   5781#  5781   5783#  5783   5785#  5785   5787#  5787   5789#  5789
5791#  5791   5793#  5793   5795#  5795   5797#  5797   5799#  5799   5801#  5801   5803#  5803
5880#  5880   5882#  5882   5884#  5884   5886#  5886   5888#  5888   5890#  5890   5892#  5892
5894#  5894   5896#  5896   5898#  5898   5901#  5901   5903#  5903   5980#  5980   5982#  5982
5984#  5984   5986#  5986   5988#  5988   6066#  6066   6068#  6068   6070#  6070   6072#  6072
6074#  6074   6076#  6076   6079#  6079   6160#  6160   6162#  6162   6164#  6164   6166#  6166
6168#  6168   6170#  6170   6172#  6172   6174#  6174   6176#  6176   6179#  6179   6181#  6181
6183#  6183   6185#  6185   6188#  6188   6265#  6265   6267#  6267   6269#  6269   6271#  6271
6273#  6273   6275#  6275   6277#  6277   6279#  6279   6281#  6281   6283#  6283   6285#  6285
6287#  6287   6289#  6289   6291#  6291   6293#  6293   6295#  6295   6373#  6373   6375#  6375
6377#  6377   6379#  6379   6455#  6455   6457#  6457   6459#  6459   6461#  6461   6463#  6463
6465#  6465   6467#  6467   6469#  6469   6471#  6471   6473#  6473   6475#  6475   6477#  6477
6479#  6479   6481#  6481   6483#  6483   6485#  6485   6487#  6487   6489#  6489   6491#  6491
6493#  6493   6495#  6495   6497#  6497   6499#  6499   6501#  6501   6503#  6503   6505#  6505
6507#  6507   6509#  6509   6511#  6511   6513#  6513   6515#  6515   6517#  6517   6519#  6519
6521#  6521   6524#  6524   6527#  6527   6602#  6602   6604#  6604   6606#  6606   6608#  6608
6610#  6610   6612#  6612   6614#  6614   6616#  6616   6618#  6618   6620#  6620   6622#  6622
6624#  6624   6626#  6626   6628#  6628   6630#  6630   6632#  6632   6634#  6634   6636#  6636
6638#  6638   6640#  6640   6642#  6642   6644#  6644   6646#  6646   6648#  6648   6650#  6650
6652#  6652   6654#  6654   6656#  6656   6658#  6658   6660#  6660   6662#  6662   6664#  6664
```

```
          6666#   6666   6669#   6669   6672#   6672   6746#   6746   6748#   6748   6750#   6750   6752#   6752
          6754#   6754   6756#   6756   6758#   6758   6760#   6760   6762#   6762   6764#   6764   6766#   6766
          6768#   6768   6770#   6770   6772#   6772   6774#   6774   6776#   6776   6778#   6778   6780#   6780
          6782#   6782   6784#   6784   6786#   6786   6788#   6788   6790#   6790   6792#   6792   6794#   6794
          6796#   6796   6798#   6798   6800#   6800   6802#   6802   6804#   6804   6806#   6806   6808#   6808
          6810#   6810   6812#   6812   6814#   6814   6816#   6816   6818#   6818   6820#   6820   6822#   6822
          6824#   6824   6826#   6826   6828#   6828   6830#   6830   6832#   6832   6834#   6834   6836#   6836
          6838#   6838   6840#   6840   6842#   6842   6845#   6845   6848#   6848   6921#   6921   6923#   6923
          6925#   6925   6927#   6927   6929#   6929   6931#   6931   6933#   6933   6935#   6935   6937#   6937
          6939#   6939   6941#   6941   6943#   6943   6945#   6945   6947#   6947   6949#   6949   6951#   6951
          6953#   6953   6955#   6955   6957#   6957   6959#   6959   6961#   6961   6963#   6963   6965#   6965
          6967#   6967   6969#   6969   6971#   6971   6973#   6973   6975#   6975   6977#   6977   6979#   6979
          6981#   6981   6983#   6983   6985#   6985   6987#   6987   6989#   6989   6991#   6991   6993#   6993
          6995#   6995   6998#   6998   7000#   7000   7002#   7002
$GARG      194#    194    196#    196    198#    198    200#    200    298#    298    300#    300    302#    302
           304#    304    306#    306    309#    309    311#    311    313#    313    315#    315    409#    409
           411#    411    413#    413    415#    415    417#    417    419#    419    421#    421    423#    423
           425#    425    427#    427    429#    429    431#    431    433#    433    436#    436    438#    438
           532#    532    534#    534    536#    536    538#    538    540#    540    542#    542    544#    544
           546#    546    548#    548    550#    550    552#    552    554#    554    556#    556    559#    559
           561#    561    665#    665    667#    667    669#    669    671#    671    673#    673    675#    675
           677#    677    679#    679    681#    681    683#    683    685#    685    687#    687    689#    689
           691#    691    693#    693    695#    695    697#    697    699#    699    701#    701    703#    703
           705#    705    707#    707    709#    709    711#    711    797#    797    799#    799    801#    801
           803#    803    888#    888    890#    890    892#    892    894#    894    979#    979    981#    981
           983#    983    985#    985   1067#   1067   1069#   1069   1071#   1071   1149#   1149   1151#   1151
          1153#   1153   1155#   1155   1157#   1157   1159#   1159   1161#   1161   1164#   1164   1240#   1240
          1242#   1242   1244#   1244   1246#   1246   1327#   1327   1329#   1329   1331#   1331   1333#   1333
          1335#   1335   1337#   1337   1339#   1339   1341#   1341   1343#   1343   1345#   1345   1347#   1347
          1349#   1349   1351#   1351   1353#   1353   1356#   1356   1358#   1358   1360#   1360   1363#   1363
          1439#   1439   1441#   1441   1443#   1443   1445#   1445   1447#   1447   1449#   1449   1451#   1451
          1453#   1453   1455#   1455   1457#   1457   1459#   1459   1461#   1461   1463#   1463   1465#   1465
          1467#   1467   1469#   1469   1471#   1471   1473#   1473   1475#   1475   1477#   1477   1479#   1479
          1481#   1481   1483#   1483   1485#   1485   1487#   1487   1489#   1489   1491#   1491   1493#   1493
          1495#   1495   1497#   1497   1499#   1499   1501#   1501   1503#   1503   1505#   1505   1507#   1507
          1509#   1509   1511#   1511   1513#   1513   1515#   1515   1517#   1517   1519#   1519   1521#   1521
          1523#   1523   1526#   1526   1528#   1528   1604#   1604   1606#   1606   1608#   1608   1610#   1610
          1612#   1612   1614#   1614   1616#   1616   1618#   1618   1620#   1620   1622#   1622   1624#   1624
          1626#   1626   1628#   1628   1630#   1630   1632#   1632   1634#   1634   1636#   1636   1638#   1638
          1640#   1640   1642#   1642   1644#   1644   1646#   1646   1648#   1648   1650#   1650   1652#   1652
          1654#   1654   1656#   1656   1658#   1658   1660#   1660   1662#   1662   1664#   1664   1666#   1666
          1668#   1668   1670#   1670   1672#   1672   1674#   1674   1676#   1676   1678#   1678   1680#   1680
          1682#   1682   1684#   1684   1686#   1686   1688#   1688   1691#   1691   1693#   1693   1769#   1769
          1771#   1771   1773#   1773   1775#   1775   1777#   1777   1779#   1779   1781#   1781   1783#   1783
          1785#   1785   1787#   1787   1789#   1789   1791#   1791   1793#   1793   1795#   1795   1797#   1797
          1799#   1799   1801#   1801   1803#   1803   1805#   1805   1807#   1807   1809#   1809   1811#   1811
          1813#   1813   1815#   1815   1817#   1817   1819#   1819   1821#   1821   1823#   1823   1825#   1625
          1827#   1827   1829#   1829   1831#   1831   1833#   1833   1835#   1835   1837#   1837   1839#   1839
          1841#   1841   1843#   1843   1845#   1845   1847#   1847   1849#   1849   1851#   1851   1853#   1853
          1855#   1855   1858#   1858   1860#   1860   1936#   1936   1938#   1938   1940#   1940   1942#   1942
          1944#   1944   1946#   1946   1948#   1948   1950#   1950   1952#   1952   1954#   1954   1956#   1956
          1958#   1958   1960#   1960   1962#   1962   1964#   1964   1966#   1966   1968#   1968   1970#   1970
          1972#   1972   1974#   1974   1976#   1976   1978#   1978   1980#   1980   1982#   1982   1984#   1984
```

```
1986#  1986   1988#  1988   1990#  1990   1992#  1992   1994#  1994   1996#  1996   1998#  1998
2000#  2000   2002#  2002   2004#  2004   2006#  2006   2008#  2008   2010#  2010   2012#  2012
2014#  2014   2016#  2016   2018#  2018   2021#  2021   2023#  2023   2099#  2099   2101#  2101
2103#  2103   2105#  2105   2107#  2107   2109#  2109   2111#  2111   2113#  2113   2115#  2115
2117#  2117   2119#  2119   2121#  2121   2123#  2123   2125#  2125   2127#  2127   2129#  2129
2131#  2131   2133#  2133   2135#  2135   2137#  2137   2139#  2139   2141#  2141   2143#  2143
2145#  2145   2147#  2147   2149#  2149   2151#  2151   2153#  2153   2155#  2155   2157#  2157
2159#  2159   2161#  2161   2163#  2163   2165#  2165   2167#  2167   2169#  2169   2171#  2171
2173#  2173   2175#  2175   2177#  2177   2179#  2179   2181#  2181   2183#  2183   2185#  2185
2188#  2188   2190#  2190   2269#  2269   2271#  2271   2273#  2273   2275#  2275   2277#  2277
2279#  2279   2281#  2281   2361#  2361   2363#  2363   2365#  2365   2367#  2367   2369#  2369
2371#  2371   2373#  2373   2375#  2375   2454#  2454   2456#  2456   2458#  2458   2460#  2460
2462#  2462   2464#  2464   2466#  2466   2468#  2468   2470#  2470   2549#  2549   2551#  2551
2553#  2553   2555#  2555   2557#  2557   2559#  2559   2561#  2561   2563#  2563   2565#  2565
2567#  2567   2646#  2646   2648#  2648   2650#  2650   2652#  2652   2654#  2654   2656#  2656
2658#  2658   2660#  2660   2662#  2662   2664#  2664   2666#  2666   2752#  2752   2754#  2754
2756#  2756   2758#  2758   2760#  2760   2762#  2762   2764#  2764   2766#  2766   2769#  2769
2771#  2771   2773#  2773   2775#  2775   2854#  2854   2856#  2856   2858#  2858   2860#  2860
2862#  2862   2864#  2864   2866#  2866   2869#  2869   2871#  2871   2873#  2873   2875#  2875
2877#  2877   2879#  2879   2881#  2881   2884#  2884   2886#  2886   2888#  2888   2890#  2890
2892#  2892   2894#  2894   2896#  2896   2899#  2899   2977#  2977   2979#  2979   2981#  2981
2983#  2983   2985#  2985   2987#  2987   2989#  2989   3064#  3064   3066#  3066   3152#  3152
3154#  3154   3156#  3156   3158#  3158   3160#  3160   3162#  3162   3164#  3164   3166#  3166
3168#  3168   3170#  3170   3252#  3252   3254#  3254   3256#  3256   3258#  3258   3260#  3260
3263#  3263   3265#  3265   3267#  3267   3269#  3269   3272#  3272   3274#  3274   3276#  3276
3279#  3279   3281#  3281   3283#  3283   3286#  3286   3288#  3288   3290#  3290   3293#  3293
3295#  3295   3297#  3297   3300#  3300   3302#  3302   3304#  3304   3307#  3307   3309#  3309
3311#  3311   3314#  3314   3316#  3316   3318#  3318   3321#  3321   3323#  3323   3325#  3325
3328#  3328   3330#  3330   3332#  3332   3335#  3335   3337#  3337   3339#  3339   3342#  3342
3344#  3344   3346#  3346   3349#  3349   3351#  3351   3353#  3353   3356#  3356   3358#  3358
3360#  3360   3363#  3363   3365#  3365   3367#  3367   3370#  3370   3372#  3372   3374#  3374
3377#  3377   3379#  3379   3381#  3381   3384#  3384   3386#  3386   3388#  3388   3391#  3391
3393#  3393   3395#  3395   3398#  3398   3400#  3400   3480#  3480   3482#  3482   3484#  3484
3486#  3486   3488#  3488   3490#  3490   3492#  3492   3494#  3494   3496#  3496   3498#  3498
3500#  3500   3503#  3503   3505#  3505   3589#  3589   3591#  3591   3593#  3593   3595#  3595
3598#  3598   3600#  3600   3602#  3602   3605#  3605   3607#  3607   3609#  3609   3612#  3612
3614#  3614   3616#  3616   3619#  3619   3621#  3621   3702#  3702   3705#  3705   3707#  3707
3709#  3709   3712#  3712   3714#  3714   3716#  3716   3719#  3719   3721#  3721   3723#  3723
3726#  3726   3728#  3728   3730#  3730   3733#  3733   3735#  3735   3738#  3738   3814#  3814
3816#  3816   3818#  3818   3820#  3820   3822#  3822   3824#  3824   3826#  3826   3828#  3828
3920#  3920   3922#  3922   3924#  3924   3926#  3926   3928#  3928   3931#  3931   3933#  3933
3935#  3935   3937#  3937   3939#  3939   3941#  3941   3943#  3943   3945#  3945   3948#  3948
4030#  4030   4032#  4032   4034#  4034   4036#  4036   4038#  4038   4040#  4040   4042#  4042
4044#  4044   4046#  4046   4048#  4048   4050#  4050   4052#  4052   4055#  4055   4136#  4136
4138#  4138   4140#  4140   4142#  4142   4144#  4144   4147#  4147   4149#  4149   4151#  4151
4238#  4238   4240#  4240   4242#  4242   4244#  4244   4246#  4246   4251#  4251   4253#  4253
4255#  4255   4257#  4257   4259#  4259   4262#  4262   4343#  4343   4345#  4345   4347#  4347
4349#  4349   4351#  4351   4353#  4353   4355#  4355   4357#  4357   4359#  4359   4361#  4361
4363#  4363   4366#  4366   4368#  4368   4370#  4370   4449#  4449   4451#  4451   4453#  4453
4455#  4455   4457#  4457   4459#  4459   4461#  4461   4463#  4463   4465#  4465   4467#  4467
4470#  4470   4473#  4473   4552#  4552   4555#  4555   4557#  4557   4559#  4559   4561#  4561
4563#  4563   4565#  4565   4567#  4567   4569#  4569   4572#  4572   4574#  4574   4576#  4576
4578#  4578   4580#  4580   4582#  4582   4584#  4584   4586#  4586   4589#  4589   4591#  4591
```

```
Q 0869    4593#  4593    4595#  4595    4597#  4597    4599#  4599    4601#  4601    4603#  4603    4606#  4606    SEQ 0882
          4608#  4608    4610#  4610    4612#  4612    4614#  4614    4616#  4616    4618#  4618    4620#  4620
          4623#  4623    4625#  4625    4627#  4627    4629#  4629    4631#  4631    4633#  4633    4635#  4635
          4637#  4637    4640#  4640    4642#  4642    4644#  4644    4646#  4646    4648#  4648    4650#  4650
          4652#  4652    4654#  4654    4657#  4657    4659#  4659    4661#  4661    4663#  4663    4665#  4665
          4667#  4667    4669#  4669    4671#  4671    4674#  4674    4676#  4676    4678#  4678    4680#  4680
          4682#  4682    4684#  4684    4686#  4686    4688#  4688    4691#  4691    4693#  4693    4695#  4695
          4697#  4697    4699#  4699    4701#  4701    4703#  4703    4705#  4705    4708#  4708    4710#  4710
          4712#  4712    4714#  4714    4716#  4716    4718#  4718    4720#  4720    4722#  4722    4725#  4725
          4727#  4727    4729#  4729    4731#  4731    4733#  4733    4735#  4735    4737#  4737    4739#  4739
          4742#  4742    4744#  4744    4746#  4746    4748#  4748    4750#  4750    4752#  4752    4754#  4754
          4756#  4756    4759#  4759    4761#  4761    4763#  4763    4765#  4765    4767#  4767    4769#  4769
          4771#  4771    4773#  4773    4775#  4775    4853#  4853    4855#  4855    4857#  4857    4859#  4859
          4861#  4861    4863#  4863    4865#  4865    4867#  4867    4869#  4869    4871#  4871    4873#  4873
          4875#  4875    4970#  4970    4972#  4972    4974#  4974    4976#  4976    4978#  4978    4980#  4980
          4982#  4982    4984#  4984    4986#  4986    4989#  4989    4991#  4991    4993#  4993    4995#  4995
          4997#  4997    4999#  4999    5001#  5001    5003#  5003    5005#  5005    5008#  5008    5010#  5010
          5012#  5012    5014#  5014    5016#  5016    5018#  5018    5020#  5020    5022#  5022    5024#  5024
          5027#  5027    5029#  5029    5031#  5031    5033#  5033    5035#  5035    5037#  5037    5039#  5039
          5041#  5041    5043#  5043    5046#  5046    5048#  5048    5050#  5050    5052#  5052    5054#  5054
          5056#  5056    5058#  5058    5060#  5060    5063#  5063    5174#  5174    5177#  5177    5180#  5180
          5183#  5183    5194#  5194    5196#  5196    5199#  5199    5201#  5201    5203#  5203    5206#  5206
          5208#  5208    5210#  5210    5213#  5213    5215#  5215    5217#  5217    5220#  5220    5222#  5222
          5224#  5224    5227#  5227    5229#  5229    5231#  5231    5234#  5234    5236#  5236    5238#  5238
          5241#  5241    5243#  5243    5245#  5245    5248#  5248    5250#  5250    5252#  5252    5255#  5255
          5257#  5257    5259#  5259    5262#  5262    5264#  5264    5266#  5266    5269#  5269    5271#  5271
          5273#  5273    5276#  5276    5278#  5278    5280#  5280    5283#  5283    5285#  5285    5287#  5287
          5290#  5290    5292#  5292    5294#  5294    5297#  5297    5299#  5299    5301#  5301    5304#  5304
          5306#  5306    5308#  5308    5311#  5311    5313#  5313    5315#  5315    5318#  5318    5320#  5320
          5322#  5322    5325#  5325    5327#  5327    5329#  5329    5332#  5332    5334#  5334    5336#  5336
          5339#  5339    5341#  5341    5343#  5343    5346#  5346    5348#  5348    5350#  5350    5353#  5353
          5355#  5355    5357#  5357    5360#  5360    5362#  5362    5364#  5364    5367#  5367    5369#  5369
          5371#  5371    5374#  5374    5376#  5376    5378#  5378    5381#  5381    5383#  5383    5386#  5386
          5388#  5388    5390#  5390    5392#  5392    5394#  5394    5396#  5396    5398#  5398    5400#  5400
          5402#  5402    5404#  5404    5406#  5406    5408#  5408    5410#  5410    5412#  5412    5414#  5414
          5416#  5416    5418#  5418    5420#  5420    5422#  5422    5424#  5424    5426#  5426    5428#  5428
          5430#  5430    5432#  5432    5434#  5434    5436#  5436    5521#  5521    5523#  5523    5525#  5525
          5527#  5527    5529#  5529    5531#  5531    5533#  5533    5535#  5535    5537#  5537    5539#  5539
          5542#  5542    5544#  5544    5546#  5546    5548#  5548    5550#  5550    5552#  5552    5554#  5554
          5634#  5634    5636#  5636    5638#  5638    5640#  5640    5642#  5642    5644#  5644    5646#  5646
          5648#  5648    5650#  5650    5652#  5652    5655#  5655    5657#  5657    5659#  5659    5661#  5661
          5663#  5663    5665#  5665    5667#  5667    5669#  5669    5671#  5671    5673#  5673    5675#  5675
          5677#  5677    5679#  5679    5681#  5681    5683#  5683    5685#  5685    5687#  5687    5689#  5689
          5691#  5691    5693#  5693    5695#  5695    5697#  5697    5699#  5699    5701#  5701    5703#  5703
          5705#  5705    5707#  5707    5709#  5709    5711#  5711    5713#  5713    5715#  5715    5717#  5717
          5719#  5719    5721#  5721    5723#  5723    5725#  5725    5727#  5727    5729#  5729    5731#  5731
          5733#  5733    5735#  5735    5737#  5737    5739#  5739    5741#  5741    5743#  5743    5745#  5745
          5747#  5747    5749#  5749    5752#  5752    5754#  5754    5757#  5757    5759#  5759    5761#  5761
          5763#  5763    5765#  5765    5767#  5767    5769#  5769    5771#  5771    5773#  5773    5775#  5775
          5777#  5777    5779#  5779    5781#  5781    5783#  5783    5785#  5785    5787#  5787    5789#  5789
          5791#  5791    5793#  5793    5795#  5795    5797#  5797    5799#  5799    5801#  5801    5803#  5803
          5880#  5880    5882#  5882    5884#  5884    5886#  5886    5888#  5888    5890#  5890    5892#  5892
          5894#  5894    5896#  5896    5898#  5898    5901#  5901    5903#  5903    5980#  5980    5982#  5982
```

```
Q 0870
        5984#  5984   5986#  5986   5988#  5988   6066#  6066   6068#  6068   6070#  6070   6072#  6072      SEQ 0883
        6074#  6074   6076#  6076   6079#  6079   6160#  6160   6162#  6162   6164#  6164   6166#  6166
        6168#  6168   6170#  6170   6172#  6172   6174#  6174   6176#  6176   6179#  6179   6181#  6181
        6183#  6183   6185#  6185   6188#  6188   6265#  6265   6267#  6267   6269#  6269   6271#  6271
        6273#  6273   6275#  6275   6277#  6277   6279#  6279   6281#  6281   6283#  6283   6285#  6285
        6287#  6287   6289#  6289   6291#  6291   6293#  6293   6295#  6295   6373#  6373   6375#  6375
        6377#  6377   6379#  6379   6455#  6455   6457#  6457   6459#  6459   6461#  6461   6463#  6463
        6465#  6465   6467#  6467   6469#  6469   6471#  6471   6473#  6473   6475#  6475   6477#  6477
        6479#  6479   6481#  6481   6483#  6483   6485#  6485   6487#  6487   6489#  6489   6491#  6491
        6493#  6493   6495#  6495   6497#  6497   6499#  6499   6501#  6501   6503#  6503   6505#  6505
        6507#  6507   6509#  6509   6511#  6511   6513#  6513   6515#  6515   6517#  6517   6519#  6519
        6521#  6521   6524#  6524   6527#  6527   6602#  6602   6604#  6604   6606#  6606   6608#  6608
        6610#  6610   6612#  6612   6614#  6614   6616#  6616   6618#  6618   6620#  6620   6622#  6622
        6624#  6624   6626#  6626   6628#  6628   6630#  6630   6632#  6632   6634#  6634   6636#  6636
        6638#  6638   6640#  6640   6642#  6642   6644#  6644   6646#  6646   6648#  6648   6650#  6650
        6652#  6652   6654#  6654   6656#  6656   6658#  6658   6660#  6660   6662#  6662   6664#  6664
        6666#  6666   6669#  6669   6672#  6672   6746#  6746   6748#  6748   6750#  6750   6752#  6752
        6754#  6754   6756#  6756   6758#  6758   6760#  6760   6762#  6762   6764#  6764   6766#  6766
        6768#  6768   6770#  6770   6772#  6772   6774#  6774   6776#  6776   6778#  6778   6780#  6780
        6782#  6782   6784#  6784   6786#  6786   6788#  6788   6790#  6790   6792#  6792   6794#  6794
        6796#  6796   6798#  6798   6800#  6800   6802#  6802   6804#  6804   6806#  6806   6808#  6808
        6810#  6810   6812#  6812   6814#  6814   6816#  6816   6818#  6818   6820#  6820   6822#  6822
        6824#  6824   6826#  6826   6828#  6828   6830#  6830   6832#  6832   6834#  6834   6836#  6836
        6838#  6838   6840#  6840   6842#  6842   6845#  6845   6848#  6848   6921#  6921   6923#  6923
        6925#  6925   6927#  6927   6929#  6929   6931#  6931   6933#  6933   6935#  6935   6937#  6937
        6939#  6939   6941#  6941   6943#  6943   6945#  6945   6947#  6947   6949#  6949   6951#  6951
        6953#  6953   6955#  6955   6957#  6957   6959#  6959   6961#  6961   6963#  6963   6965#  6965
        6967#  6967   6969#  6969   6971#  6971   6973#  6973   6975#  6975   6977#  6977   6979#  6979
        6981#  6981   6983#  6983   6985#  6985   6987#  6987   6989#  6989   6991#  6991   6993#  6993
        6995#  6995   6998#  6998   7000#  7000   7002#  7002
%ADDR    194#   194    196    196#   198    198#   200    200#   298#   298    300    300#   302    302#
         304    304#   306    306#   309#   309    311#   311    313#   313    315#   315    409#   409
         411#   411    413    413#   415    415#   417    417#   419    419#   421    421#   423    423#
         425    425#   427    427#   429    429#   431    431#   433    433#   436#   436    438#   438
         532#   532    534#   534    536    536#   538    538#   540    540#   542    542#   544    544#
         546    546#   548    548#   550    550#   552    552#   554    554#   556    556#   559#   559
         561#   561    665#   665    667    667#   669    669#   671    671#   673    673#   675#   675
         677    677#   679#   679    681    681#   683#   683    685    685#   687#   687    689    689#
         691#   691    693    693#   695#   695    697    697#   699#   699    701    701#   703#   703
         705    705#   707#   707    709    709#   711#   711    797#   797    799#   799    801#   801
         803#   803    888#   888    890#   890    892#   892    894#   894    979#   979    981#   981
         983#   983    985#   985   1067#  1067   1069#  1069   1071#  1071   1149#  1149   1151   1151#
        1153   1153#  1155   1155#  1157   1157#  1159   1159#  1161   1161#  1164#  1164   1240#  1240
        1242   1242#  1244   1244#  1246#  1246   1327#  1327   1329   1329#  1331   1331#  1333   1333#
        1335   1335#    37   1337#  1339   1339#  1341   1341#  1343   1343#  1345   1345#  1347   1347#
        1349   1349#   351   1351#  1353   1353#  1356#  1356   1358   1358#  1360   1360#  1363#  1363
        1439#  1439    441   1441#  1443   1443#  1445   1445#  1447   1447#  1449#  1449   1451#  1451
        1453#  1453   1455#  1455   1457#  1457   1459#  1459   1461#  1461   1463#  1463   1465#  1465
        1467#  1467   1469#  1469   1471#  1471   1473#  1473   1475#  1475   1477#  1477   1479#  1479
        1481#  1481   1483#  1483   1485#  1485   1487#  1487   1489#  1489   1491#  1491   1493#  1493
        1495#  1495   1497#  1497   1499#  1499   1501#  1501   1503#  1503   1505#  1505   1507#  1507
        1509#  1509   1511#  1511   1513#  1513   1515#  1515   1517#  1517   1519#  1519   1521#  1521
        1523#  1523   1526#  1526   1528#  1528   1604#  1604   1606   1606#  1608   1608#  1610   1610#
```

EQ 0871

```
1612    1612#   1614#   1614    1616#   1616    1618#   1618    1620#   1620    1622#   1622    1624#   1624
1626#   1626    1628#   1628    1630#   1630    1632#   1632    1634#   1634    1636#   1636    1638#   1638
1640#   1640    1642#   1642    1644#   1644    1646#   1646    1648#   1648    1650#   1650    1652#   1652
1654#   1654    1656#   1656    1658#   1658    1660#   1660    1662#   1662    1664#   1664    1666#   1666
1668#   1668    1670#   1670    1672#   1672    1674#   1674    1676#   1676    1678#   1678    1680#   1680
1682#   1682    1684#   1684    1686#   1686    1688#   1688    1691#   1691    1693#   1693    1769#   1769
1771    1771#   1773    1773#   1775    1775#   1777    1777#   1779    1779#   1781#   1781    1783#   1783
1785#   1785    1787#   1787    1789#   1789    1791#   1791    1793#   1793    1795#   1795    1797#   1797
1799#   1799    1801#   1801    1803#   1803    1805#   1805    1807#   1807    1809#   1809    1811#   1811
1813#   1813    1815#   1815    1817#   1817    1819#   1819    1821#   1821    1823#   1823    1825#   1825
1827#   1827    1829#   1829    1831#   1831    1833#   1833    1835#   1835    1837#   1837    1839#   1839
1841#   1841    1843#   1843    1845#   1845    1847#   1847    1849#   1849    1851#   1851    1853#   1853
1855#   1855    1858#   1858    1860#   1860    1936#   1936    1938    1938#   1940    1940#   1942    1942#
1944    1944#   1946    1946#   1948#   1948    1950#   1950    1952#   1952    1954#   1954    1956#   1956
1958#   1958    1960#   1960    1962#   1962    1964#   1964    1966#   1966    1968#   1968    1970#   1970
1972#   1972    1974#   1974    1976#   1976    1978#   1978    1980#   1980    1982#   1982    1984#   1984
1986#   1986    1988#   1988    1990#   1990    1992#   1992    1994#   1994    1996#   1996    1998#   1998
2000#   2000    2002#   2002    2004#   2004    2006#   2006    2008#   2008    2010#   2010    2012#   2012
2014#   2014    2016#   2016    2018#   2018    2021#   2021    2023#   2023    2099#   2099    2101    2101#
2103    2103#   2105    2105#   2107    2107#   2109    2109#   2111#   2111    2113#   2113    2115#   2115
2117#   2117    2119#   2119    2121#   2121    2123#   2123    2125#   2125    2127#   2127    2129#   2129
2131#   2131    2133#   2133    2135#   2135    2137#   2137    2139#   2139    2141#   2141    2143#   2143
2145#   2145    2147#   2147    2149#   2149    2151#   2151    2153#   2153    2155#   2155    2157#   2157
2159#   2159    2161#   2161    2163#   2163    2165#   2165    2167#   2167    2169#   2169    2171#   2171
2173#   2173    2175#   2175    2177#   2177    2179#   2179    2181#   2181    2183#   2183    2185#   2185
2188#   2188    2190#   2190    2269#   2269    2271    2271#   2273#   2273    2275#   2275    2277#   2277
2279#   2279    2281#   2281    2361#   2361    2363    2363#   2365#   2365    2367#   2367    2369#   2369
2371#   2371    2373#   2373    2375#   2375    2454#   2454    2456    2456#   2458#   2458    2460#   2460
2462#   2462    2464#   2464    2466#   2466    2468#   2468    2470#   2470    2549#   2549    2551    2551#
2553#   2553    2555#   2555    2557#   2557    2559#   2559    2561#   2561    2563#   2563    2565#   2565
2567#   2567    2646#   2646    2648    2648#   2650#   2650    2652#   2652    2654#   2654    2656#   2656
2658#   2658    2660#   2660    2662#   2662    2664#   2664    2666#   2666    2752#   2752    2754    2754#
2756    2756#   2758    2758#   2760    2760#   2762    2762#   2764    2764#   2766    2766#   2769#   2769
2771#   2771    2773#   2773    2775#   2775    2854#   2854    2856#   2856    2858#   2858    2860#   2860
2862#   2862    2864#   2864    2866#   2866    2869#   2869    2871#   2871    2873#   2873    2875#   2875
2877#   2877    2879#   2879    2881#   2881    2884#   2884    2886#   2886    2888#   2888    2890#   2890
2892#   2892    2894#   2894    2896#   2896    2899#   2899    2977#   2977    2979#   2979    2981#   2981
2983#   2983    2985#   2985    2987#   2987    2989#   2989    3064#   3064    3066    3066#   3152#   3152
3154#   3154    3156    3156#   3158    3158#   3160#   3160    3162    3162#   3164    3164#   3166#   3166
3168    3168#   3170    3170#   3252#   3252    3254#   3254    3256    3256#   3258    3258#   3260#   3260
3263#   3263    3265    3265#   3267    3267#   3269#   3269    3272#   3272    3274    3274#   3276    3276#
3279#   3279    3281    3281#   3283    3283#   3286#   3286    3288    3288#   3290    3290#   3293#   3293
3295    3295#   3297    3297#   3300#   3300    3302    3302#   3304    3304#   3307#   3307    3309    3309#
3311    3311#   3314#   3314    3316    3316#   3318    3318#   3321#   3321    3323    3323#   3325    3325#
3328#   3328    3330    3330#   3332    3332#   3335#   3335    3337    3337#   3339    3339#   3342#   3342
3344    3344#   3346    3346#   3349#   3349    3351    3351#   3353    3353#   3356#   3356    3358    3358#
3360    3360#   3363#   3363    3365    3365#   3367    3367#   3370#   3370    3372    3372#   3374    3374#
3377#   3377    3379    3379#   3381    3381#   3384#   3384    3386    3386#   3388    3388#   3391#   3391
3393    3393#   3395    3395#   3398    3398#   3400    3400#   3480#   3480    3482    3482#   3484    3484#
3486    3486#   3488    3488#   3490    3490#   3492    3492#   3494    3494#   3496    3496#   3498    3498#
3500    3500#   3503#   3503    3505#   3505    3589#   3589    3591#   3591    3593    3593#   3595    3595#
3598    3598#   3600    3600#   3602    3602#   3605    3605#   3607    3607#   3609    3609#   3612    3612#
3614    3614#   3616    3616#   3619    3619#   3621    3621#   3702#   3702    3705    3705#   3707    3707#
```

SEQ 0884

```
3709       3709#      3712       3712#      3714       3714#      3716       3716#      3719       3719#      3721       3721#      3723       3723#
3726       3726#      3728       3728#      3730       3730#      3733       3733#      3735#      3735       3738#      3738       3814#      3814
3816#      3816       3818       3818#      3820       3820#      3822       3822#      3824       3824#      3826       3826#      3828       3828#
3920#      3920       3922       3922#      3924       3924#      3926       3926#      3928       3928#      3931       3931#      3933       3933#
3935       3935#      3937       3937#      3939       3939#      3941       3941#      3943       3943#      3945       3945#      3948#      3948
4030#      4030       4032#      4032       4034       4034#      4036       4036#      4038       4038#      4040       4040#      4042       4042#
4044       4044#      4046       4046#      4048       4048#      4050       4050#      4052       4052#      4055#      4055       4136#      4136
4138#      4138       4140       4140#      4142       4142#      4144       4144#      4147       4147#      4149       4149#      4151       4151#
4238#      4238       4240#      4240       4242       4242#      4244       4244#      4246       4246#      4251#      4251       4253#      4253
4255       4255#      4257       4257#      4259       4259#      4262       4262#      4343#      4343       4345       4345#      4347       4347#
4349       4349#      4351       4351#      4353       4353#      4355       4355#      4357       4357#      4359       4359#      4361       4361#
4363       4363#      4366#      4366       4368       4368#      4370       4370#      4449#      4449       4451       4451#      4453       4453#
4455       4455#      4457       4457#      4459       4459#      4461       4461#      4463       4463#      4465       4465#      4467       4467#
4470#      4470       4473#      4473       4552#      4552       4555#      4555       4557       4557#      4559       4559#      4561       4561#
4563       4563#      4565       4565#      4567       4567#      4569       4569#      4572       4572#      4574       4574#      4576       4576#
4578       4578#      4580       4580#      4582       4582#      4584       4584#      4586       4586#      4589       4589#      4591       4591#
4593       4593#      4595       4595#      4597       4597#      4599       4599#      4601       4601#      4603       4603#      4606       4606#
4608       4608#      4610       4610#      4612       4612#      4614       4614#      4616       4616#      4618       4618#      4620       4620#
4623       4623#      4625       4625#      4627       4627#      4629       4629#      4631       4631#      4633       4633#      4635       4635#
4637       4637#      4640       4640#      4642       4642#      4644       4644#      4646       4646#      4648       4648#      4650       4650#
4652       4652#      4654       4654#      4657       4657#      4659       4659#      4661       4661#      4663       4663#      4665       4665#
4667       4667#      4669       4669#      4671       4671#      4674       4674#      4676       4676#      4678       4678#      4680       4680#
4682       4682#      4684       4684#      4686       4686#      4688       4688#      4691       4691#      4693       4693#      4695       4695#
4697       4697#      4699       4699#      4701       4701#      4703       4703#      4705       4705#      4708       4708#      4710       4710#
4712       4712#      4714       4714#      4716       4716#      4718       4718#      4720       4720#      4722       4722#      4725       4725#
4727       4727#      4729       4729#      4731       4731#      4733       4733#      4735       4735#      4737       4737#      4739       4739#
4742       4742#      4744       4744#      4746       4746#      4748       4748#      4750       4750#      4752       4752#      4754       4754#
4756       4756#      4759#      4759       4761#      4761       4763       4763#      4765       4765#      4767       4767#      4769       4769#
4771       4771#      4773       4773#      4775       4775#      4853#      4853       4855       4855#      4857       4857#      4859       4859#
4861       4861#      4863       4863#      4865       4865#      4867       4867#      4869       4869#      4871       4871#      4873#      4873
4875       4875#      4970#      4970       4972       4972#      4974       4974#      4976       4976#      4978       4978#      4980       4980#
4982       4982#      4984       4984#      4986       4986#      4989#      4989       4991       4991#      4993       4993#      4995       4995#
4997       4997#      4999       4999#      5001       5001#      5003       5003#      5005       5005#      5008#      5008       5010       5010#
5012       5012#      5014       5014#      5016       5016#      5018       5018#      5020       5020#      5022       5022#      5024       5024#
5027#      5027       5029       5029#      5031       5031#      5033       5033#      5035       5035#      5037       5037#      5039       5039#
5041       5041#      5043       5043#      5046#      5046       5048       5048#      5050       5050#      5052       5052#      5054       5054#
5056       5056#      5058       5058#      5060       5060#      5063#      5063       5174#      5174       5177#      5177       5180#      5180
5183#      5183       5194#      5194       5196#      5196       5199#      5199       5201       5201#      5203       5203#      5206       5206#
5208       5208#      5210       5210#      5213       5213#      5215       5215#      5217       5217#      5220       5220#      5222       5222#
5224       5224#      5227       5227#      5229       5229#      5231       5231#      5234       5234#      5236       5236#      5238       5238#
5241       5241#      5243       5243#      5245       5245#      5248       5248#      5250       5250#      5252       5252#      5255       5255#
5257       5257#      5259       5259#      5262       5262#      5264       5264#      5266       5266#      5269       5269#      5271       5271#
5273       5273#      5276       5276#      5278       5278#      5280       5280#      5283       5283#      5285       5285#      5287       5287#
5290       5290#      5292       5292#      5294       5294#      5297       5297#      5299       5299#      5301       5301#      5304       5304#
5306       5306#      5308       5308#      5311       5311#      5313       5313#      5315       5315#      5318       5318#      5320       5320#
5322       5322#      5325       5325#      5327       5327#      5329       5329#      5332       5332#      5334       5334#      5336       5336#
5339       5339#      5341       5341#      5343       5343#      5346       5346#      5348       5348#      5350       5350#      5353       5353#
5355       5355#      5357       5357#      5360       5360#      5362       5362#      5364       5364#      5367       5367#      5369       5369#
5371       5371#      5374       5374#      5376       5376#      5378       5378#      5381       5381#      5383       5383#      5386#      5386
5388#      5388       5390#      5390       5392#      5392       5394#      5394       5396#      5396       5398#      5398       5400#      5400
5402#      5402       5404#      5404       5406#      5406       5408#      5408       5410#      5410       5412#      5412       5414#      5414
5416#      5416       5418#      5418       5420#      5420       5422#      5422       5424#      5424       5426#      5426       5428#      5428
5430#      5430       5432#      5432       5434#      5434       5436#      5436       5521#      5521       5523#      5523       5525       5525#
```

```
        5527    5527#   5529    5529#   5531    5531#   5533    5533#   5535    5535#   5537    5537#   5539    5539#     SEQ 0886
        5542    5542#   5544    5544#   5546    5546#   5548    5548#   5550    5550#   5552    5552#   5554#   5554
        5634#   5634    5636#   5636    5638    5638#   5640    5640#   5642    5642#   5644    5644#   5646    5646#
        5648    5648#   5650    5650#   5652    5652#   5655#   5655    5657    5657#   5659    5659#   5661    5661#
        5663    5663#   5665    5665#   5667    5667#   5669    5669#   5671    5671#   5673    5673#   5675    5675#
        5677    5677#   5679    5679#   5681    5681#   5683    5683#   5685    5685#   5687    5687#   5689    5689#
        5691    5691#   5693    5693#   5695    5695#   5697    5697#   5699    5699#   5701    5701#   5703    5703#
        5705    5705#   5707    5707#   5709    5709#   5711    5711#   5713    5713#   5715    5715#   5717    5717#
        5719    5719#   5721    5721#   5723    5723#   5725    5725#   5727    5727#   5729    5729#   5731    5731#
        5733    5733#   5735    5735#   5737    5737#   5739    5739#   5741    5741#   5743    5743#   5745    5745#
        5747    5747#   5749    5749#   5752    5752#   5754    5754#   5757#   5757    5759#   5759    5761#   5761
        5763#   5763    5765#   5765    5767#   5767    5769#   5769    5771#   5771    5773#   5773    5775#   5775
        5777#   5777    5779#   5779    5781#   5781    5783#   5783    5785#   5785    5787#   5787    5789#   5789
        5791#   5791    5793#   5793    5795#   5795    5797#   5797    5799#   5799    5801#   5801    5803#   5803
        5880#   5880    5882#   5882    5884    5884#   5886    5886#   5888    5888#   5890    5890#   5892    5892#
        5894    5894#   5896    5896#   5898    5898#   5901#   5901    5903#   5903    5980#   5980    5982#   5982
        5984    5984#   5986    5986#   5988    5988#   6056#   6066    6068    6068#   6070    6070#   6072#   6072
        6074    6074#   6076    6076#   6079#   6079    6160#   6160    6162    6162#   6164    6164#   6166    6166#
        6168    6168#   6170    6170#   6172    6172#   6174    6174#   6176    6176#   6179    6179#   6181    6181#
        6183    6183#   6185    6185#   6188#   6188    6265#   6265    6267    6267#   6269    6269#   6271    6271#
        6273    6273#   6275    6275#   6277    6277#   6279    6279#   6281    6281#   6283    6283#   6285    6285#
        6287    6287#   6289    6289#   6291    6291#   6293    6293#   6295#   6295    6373#   6373    6375    6375#
        6377    6377#   6379#   6379    6455#   6455    6457#   6457    6459    6459#   6461    6461#   6463    6463#
        6465    6465#   6467    6467#   6469    6469#   6471    6471#   6473    6473#   6475    6475#   6477    6477#
        6479    6479#   6481    6481#   6483    6483#   6485    6485#   6487    6487#   6489    6489#   6491    6491#
        6493    6493#   6495    6495#   6497    6497#   6499    6499#   6501    6501#   6503    6503#   6505    6505#
        6507    6507#   6509    6509#   6511    6511#   6513    6513#   6515    6515#   6517    6517#   6519    6519#
        6521    6521#   6524    6524#   6527#   6527    6602#   6602    6604#   6604    6606    6606#   6608    6608#
        6610    6610#   6612    6612#   6614    6614#   6616    6616#   6618    6618#   6620    6620#   6622    6622#
        6624    6624#   6626    6626#   6628    6628#   6630    6630#   6632    6632#   6634    6634#   6636    6636#
        6638    6638#   6640    6640#   6642    6642#   6644    6644#   6646    6646#   6648    6648#   6650    6650#
        6652    6652#   6654    6654#   6656    6656#   6658    6658#   6660    6660#   6662    6662#   6664    6664#
        6666    6666#   6669    6669#   6672#   6672    6746#   6746    6748#   6748    6750    6750#   6752    6752#
        6754    6754#   6756    6756#   6758    6758#   6760    6760#   6762    6762#   6764    6764#   6766    6766#
        6768    6768#   6770    6770#   6772    6772#   6774    6774#   6776    6776#   6778    6778#   6780    6780#
        6782    6782#   6784    6784#   6786    6786#   6788    6788#   6790    6790#   6792    6792#   6794    6794#
        6796    6796#   6798    6798#   6800    6800#   6802    6802#   6804    6804#   6806    6806#   6808    6808#
        6810    6810#   6812    6812#   6814    6814#   6816    6816#   6818    6818#   6820    6820#   6822    6822#
        6824    6824#   6826    6826#   6828    6828#   6830    6830#   6832    6832#   6834    6834#   6836    6836#
        6838    6838#   6840    6840#   6842    6842#   6845    6845#   6848#   6848    6921#   6921    6923#   6923
        6925    6925#   6927    6927#   6929    6929#   6931    6931#   6933    6933#   6935    6935#   6937    6937#
        6939    6939#   6941    6941#   6943    6943#   6945    6945#   6947    6947#   6949    6949#   6951    6951#
        6953    6953#   6955    6955#   6957    6957#   6959    6959#   6961    6961#   6963    6963#   6965    6965#
        6967    6967#   6969    6969#   6971    6971#   6973    6973#   6975    6975#   6977    6977#   6979    6979#
        6981    6981#   6983    6983#   6985    6985#   6987    6987#   6989    6989#   6991    6991#   6993    6993#
        6995    6995#   6998#   6998    7000    7000#   7002#   7002
%ML      194#    194     196#    196     198#    198     200#    200     298#    298     300#    300     302#    302
         304#    304     306#    306     309#    309     311#    311     313#    313     315#    315     409#    409
         411#    411     413#    413     415#    415     417#    417     419#    419     421#    421     423#    423
         425#    425     427#    427     429#    429     431#    431     433#    433     436#    436     438#    438
         532#    532     534#    534     536#    536     538#    538     540#    540     542#    542     544#    544
         546#    546     548#    548     550#    550     552#    552     554#    554     556#    556     559#    559
         561#    561     665#    665     667#    667     669#    669     671#    671     673#    673     675#    675
```

```
 677#  677   679#  679   681#  681   683#  683   685#  685   687#  687   689#  689     SEQ 0887
 691#  691   693#  693   695#  695   697#  697   699#  699   701#  701   703#  703
 705#  705   707#  707   709#  709   711#  711   797#  797   799#  799   801#  801
 803#  803   888#  888   890#  890   892#  892   894#  894   979#  979   981#  981
 983#  983   985#  985  1067# 1067  1069# 1069  1071# 1071  1149# 1149  1151# 1151
1153# 1153  1155# 1155  1157# 1157  1159# 1159  1161# 1161  1164# 1164  1240# 1240
1242# 1242  1244# 1244  1246# 1246  1327# 1327  1329# 1329  1331# 1331  1333# 1333
1335# 1335  1337# 1337  1339# 1339  1341# 1341  1343# 1343  1345# 1345  1347# 1347
1349# 1349  1351# 1351  1353# 1353  1356# 1356  1358# 1358  1360# 1360  1363# 1363
1439# 1439  1441# 1441  1443# 1443  1445# 1445  1447# 1447  1449# 1449  1451# 1451
1453# 1453  1455# 1455  1457# 1457  1459# 1459  1461# 1461  1463# 1463  1465# 1465
1467# 1467  1469# 1469  1471# 1471  1473# 1473  1475# 1475  1477# 1477  1479# 1479
1481# 1481  1483# 1483  1485# 1485  1487# 1487  1489# 1489  1491# 1491  1493# 1493
1495# 1495  1497# 1497  1499# 1499  1501# 1501  1503# 1503  1505# 1505  1507# 1507
1509# 1509  1511# 1511  1513# 1513  1515# 1515  1517# 1517  1519# 1519  1521# 1521
1523# 1523  1526# 1526  1528# 1528  1604# 1604  1606# 1606  1608# 1608  1610# 1610
1612# 1612  1614# 1614  1616# 1616  1618# 1618  1620# 1620  1622# 1622  1624# 1624
1626# 1626  1628# 1628  1630# 1630  1632# 1632  1634# 1634  1636# 1636  1638# 1638
1640# 1640  1642# 1642  1644# 1644  1646# 1646  1648# 1648  1650# 1650  1652# 1652
1654# 1654  1656# 1656  1658# 1658  1660# 1660  1662# 1662  1664# 1664  1666# 1666
1668# 1668  1670# 1670  1672# 1672  1674# 1674  1676# 1676  1678# 1678  1680# 1680
1682# 1682  1684# 1684  1686# 1686  1688# 1688  1691# 1691  1693# 1693  1769# 1769
1771# 1771  1773# 1773  1775# 1775  1777# 1777  1779# 1779  1781# 1781  1783# 1783
1785# 1785  1787# 1787  1789# 1789  1791# 1791  1793# 1793  1795# 1795  1797# 1797
1799# 1799  1801# 1801  1803# 1803  1805# 1805  1807# 1807  1809# 1809  1811# 1811
1813# 1813  1815# 1815  1817# 1817  1819# 1819  1821# 1821  1823# 1823  1825# 1825
1827# 1827  1829# 1829  1831# 1831  1833# 1833  1835# 1835  1837# 1837  1839# 1839
1841# 1841  1843# 1843  1845# 1845  1847# 1847  1849# 1849  1851# 1851  1853# 1853
1855# 1855  1858# 1858  1860# 1860  1936# 1936  1938# 1938  1940# 1940  1942# 1942
1944# 1944  1946# 1946  1948# 1948  1950# 1950  1952# 1952  1954# 1954  1956# 1956
1958# 1958  1960# 1960  1962# 1962  1964# 1964  1966# 1966  1968# 1968  1970# 1970
1972# 1972  1974# 1974  1976# 1976  1978# 1978  1980# 1980  1982# 1982  1984# 1984
1986# 1986  1988# 1988  1990# 1990  1992# 1992  1994# 1994  1996# 1996  1998# 1998
2000# 2000  2002# 2002  2004# 2004  2006# 2006  2008# 2008  2010# 2010  2012# 2012
2014# 2014  2016# 2016  2018# 2018  2021# 2021  2023# 2023  2099# 2099  2101# 2101
2103# 2103  2105# 2105  2107# 2107  2109# 2109  2111# 2111  2113# 2113  2115# 2115
2117# 2117  2119# 2119  2121# 2121  2123# 2123  2125# 2125  2127# 2127  2129# 2129
2131# 2131  2133# 2133  2135# 2135  2137# 2137  2139# 2139  2141# 2141  2143# 2143
2145# 2145  2147# 2147  2149# 2149  2151# 2151  2153# 2153  2155# 2155  2157# 2157
2159# 2159  2161# 2161  2163# [illegible]  2165# 2165  2167# 2167  2169# 2169  2171# 2171
2173# 2173  2175# 2175  2177# 2177  2179# 2179  2181# 2181  2183# 2183  2185# 2185
2188# 2188  2190# 2190  2269# 2269  2271# 2271  2273# 2273  2275# 2275  2277# 2277
2279# 2279  2281# 2281  2361# 2361  2363# 2363  2365# 2365  2367# 2367  2369# 2369
2371# 2371  2373# 2373  2375# 2375  2454# 2454  2456# 2456  2458# 2458  2460# 2460
2462# 2462  2464# 2464  2466# 2466  2468# 2468  2470# 2470  2549# 2549  2551# 2551
2553# 2553  2555# 2555  2557# 2557  2559# 2559  2561# 2561  2563# 2563  2565# 2565
2567# 2567  2646# 2646  2648# 2648  2650# 2650  2652# 2652  2654# 2654  2656# 2656
2658# 2658  2660# 2660  2662# 2662  2664# 2664  2666# 2666  2752# 2752  2754# 2754
2756# 2756  2758# 2758  2760# 2760  2762# 2762  2764# 2764  2766# 2766  2769# 2769
2771# 2771  2773# 2773  2775# 2775  2854# 2854  2856# 2856  2858# 2858  2860# 2860
2862# 2862  2864# 2864  2866# 2866  2869# 2869  2871# 2871  2873# 2873  2875# 2875
2877# 2877  2879# 2879  2881# 2881  2884# 2884  2886# 2886  2888# 2888  2890# 2890
2892# 2892  2894# 2894  2896# 2896  2899# 2899  2977# 2977  2979# 2979  2981# 2981
```

```
2983#   2983     2985#   2985     2987#   2987     2989#   2989     3064#   3064     3066#   3066     3152#   3152
3154#   3154     3156#   3156     3158#   3158     3160#   3160     3162#   3162     3164#   3164     3166#   3166         SEQ 0888
3168#   3168     3170#   3170     3252#   3252     3254#   3254     3256#   3256     3258#   3258     3260#   3260
3263#   3263     3265#   3265     3267#   3267     3269#   3269     3272#   3272     3274#   3274     3276#   3276
3279#   3279     3281#   3281     3283#   3283     3286#   3286     3288#   3288     3290#   3290     3293#   3293
3295#   3295     3297#   3297     3300#   3300     3302#   3302     3304#   3304     3307#   3307     3309#   3309
3311#   3311     3314#   3314     3316#   3316     3318#   3318     3321#   3321     3323#   3323     3325#   3325
3328#   3328     3330#   3330     3332#   3332     3335#   3335     3337#   3337     3339#   3339     3342#   3342
3344#   3344     3346#   3346     3349#   3349     3351#   3351     3353#   3353     3356#   3356     3358#   3358
3360#   3360     3363#   3363     3365#   3365     3367#   3367     3370#   3370     3372#   3372     3374#   3374
3377#   3377     3379#   3379     3381#   3381     3384#   3384     3386#   3386     3388#   3388     3391#   3391
3393#   3393     3395#   3395     3398#   3398     3400#   3400     3480#   3480     3482#   3482     3484#   3484
3486#   3486     3488#   3488     3490#   3490     3492#   3492     3494#   3494     3496#   3496     3498#   3498
3500#   3500     3503#   3503     3505#   3505     3589#   3589     3591#   3591     3593#   3593     3595#   3595
3598#   3598     3600#   3600     3602#   3602     3605#   3605     3607#   3607     3609#   3609     3612#   3612
3614#   3614     3616#   3616     3619#   3619     3621#   3621     3702#   3702     3705#   3705     3707#   3707
3709#   3709     3712#   3712     3714#   3714     3716#   3716     3719#   3719     3721#   3721     3723#   3723
3726#   3726     3728#   3728     3730#   3730     3733#   3733     3735#   3735     3738#   3738     3814#   3814
3816#   3816     3818#   3818     3820#   3820     3822#   3822     3824#   3824     3826#   3826     3828#   3828
3920#   3920     3922#   3922     3924#   3924     3926#   3926     3928#   3928     3931#   3931     3933#   3933
3935#   3935     3937#   3937     3939#   3939     3941#   3941     3943#   3943     3945#   3945     3948#   3948
4030#   4030     4032#   4032     4034#   4034     4036#   4036     4038#   4038     4040#   4040     4042#   4042
4044#   5044     4046#   4046     4048#   4048     4050#   4050     4052#   4052     4055#   4055     4136#   4136
4138#   4138     4140#   4140     4142#   4142     4144#   4144     4147#   4147     4149#   4149     4151#   4151
4238#   4238     4240#   4240     4242#   4242     4244#   4244     4246#   4246     4251#   4251     4253#   4253
4255#   4255     4257#   4257     4259#   4259     4262#   4262     4343#   4343     4345#   4345     4347#   4347
4349#   4349     4351#   4351     4353#   4353     4355#   4355     4357#   4357     4359#   4359     4361#   4361
4363#   4363     4366#   4366     4368#   4368     4370#   4370     4449#   4449     4451#   4451     4453#   4453
4455#   4455     4457#   4457     4459#   4459     4461#   4461     4463#   4463     4465#   4465     4467#   4467
4470#   4470     4473#   4473     4552#   4552     4555#   4555     4557#   4557     4559#   4559     4561#   4561
4563#   4563     4565#   4565     4567#   4567     4569#   4569     4572#   4572     4574#   4574     4576#   4576
4578#   4578     4580#   4580     4582#   4582     4584#   4584     4586#   4586     4589#   4589     4591#   4591
4593#   4593     4595#   4595     4597#   4597     4599#   4599     4601#   4601     4603#   4603     4606#   4606
4608#   4608     4610#   4610     4612#   4612     4614#   4614     4616#   4616     4618#   4618     4620#   4620
4623#   4623     4625#   4625     4627#   4627     4629#   4629     4631#   4631     4633#   4633     4635#   4635
4637#   4637     4640#   4640     4642#   4642     4644#   4644     4646#   4646     4648#   4648     4650#   4650
4652#   4652     4654#   4654     4657#   4657     4659#   4659     4661#   4661     4663#   4663     4665#   4665
4667#   4667     4669#   4669     4671#   4671     4674#   4674     4676#   4676     4678#   4678     4680#   4680
4682#   4682     4684#   4684     4686#   4686     4688#   4688     4691#   4691     4693#   4693     4695#   4695
4697#   4697     4699#   4699     4701#   4701     4703#   4703     4705#   4705     4708#   4708     4710#   4710
4712#   4712     4714#   4714     4716#   4716     4718#   4718     4720#   4720     4722#   4722     4725#   4725
4727#   4727     4729#   4729     4731#   4731     4733#   4733     4735#   4735     4737#   4737     4739#   4739
4742#   4742     4744#   4744     4746#   4746     4748#   4748     4750#   4750     4752#   4752     4754#   4754
4756#   4756     4759#   4759     4761#   4761     4763#   4763     4765#   4765     4767#   4767     4769#   4769
4771#   4771     4773#   4773     4775#   4775     4853#   4853     4855#   4855     4857#   4857     4859#   4859
4861#   4861     4863#   4863     4865#   4865     4867#   4867     4869#   4869     4871#   4871     4873#   4873
4875#   4875     4970#   4970     4972#   4972     4974#   4974     4976#   4976     4978#   4978     4980#   4980
4982#   4982     4984#   4984     4986#   4986     4989#   4989     4991#   4991     4993#   4993     4995#   4995
4997#   4997     4999#   4999     5001#   5001     5003#   5003     5005#   5005     5008#   5008     5010#   5010
5012#   5012     5014#   5014     5016#   5016     5018#   5018     5020#   5020     5022#   5022     5024#   5024
5027#   5027     5029#   5029     5031#   5031     5033#   5033     5035#   5035     5037#   5037     5039#   5039
5041#   5041     5043#   5043     5046#   5046     5048#   5048     5050#   5050     5052#   5052     5054#   5054
5056#   5056     5058#   5058     5060#   5060     5063#   5063     5174#   5174     5177#   5177     5180#   5180
```

```
5183#    5183       5194#    5194       5196#    5196       5199#    5199       5201#    5201       5203#    5203       5206#    5206          SEQ 0889
5208#    5208       5210#    5210       5213#    5213       5215#    5215       5217#    5217       5220#    5220       5222#    5222
5224#    5224       5227#    5227       5229#    5229       5231#    5231       5234#    5234       5236#    5236       5238#    5238
5241#    5241       5243#    5243       5245#    5245       5248#    5248       5250#    5250       5252#    5252       5255#    5255
5257#    5257       5259#    5259       5262#    5262       5264#    5264       5266#    5266       5269#    5269       5271#    5271
5273#    5273       5276#    5276       5278#    5278       5280#    5280       5283#    5283       5285#    5285       5287#    5287
5290#    5290       5292#    5292       5294#    5294       5297#    5297       5299#    5299       5301#    5301       5304#    5304
5306#    5306       5308#    5308       5311#    5311       5313#    5313       5315#    5315       5318#    5318       5320#    5320
5322#    5322       5325#    5325       5327#    5327       5329#    5329       5332#    5332       5334#    5334       5336#    5336
5339#    5339       5341#    5341       5343#    5343       5346#    5346       5348#    5348       5350#    5350       5353#    5353
5355#    5355       5357#    5357       5360#    5360       5362#    5362       5364#    5364       5367#    5367       5369#    5369
5371#    5371       5374#    5374       5376#    5376       5378#    5378       5381#    5381       5383#    5383       5386#    5386
5388#    5388       5390#    5390       5392#    5392       5394#    5394       5396#    5396       5398#    5398       5400#    5400
5402#    5402       5404#    5404       5406#    5406       5408#    5408       5410#    5410       5412#    5412       5414#    5414
5416#    5416       5418#    5418       5420#    5420       5422#    5422       5424#    5424       5426#    5426       5428#    5428
5430#    5430       5432#    5432       5434#    5434       5436#    5436       5521#    5521       5523#    5523       5525#    5525
5527#    5527       5529#    5529       5531#    5531       5533#    5533       5535#    5535       5537#    5537       5539#    5539
5542#    5542       5544#    5544       5546#    5546       5548#    5548       5550#    5550       5552#    5552       5554#    5554
5634#    5634       5636#    5636       5638#    5638       5640#    5640       5642#    5642       5644#    5644       5646#    5646
5648#    5648       5650#    5650       5652#    5652       5655#    5655       5657#    5657       5659#    5659       5661#    5661
5663#    5663       5665#    5665       5667#    5667       5669#    5669       5671#    5671       5673#    5673       5675#    5675
5677#    5677       5679#    5679       5681#    5681       5683#    5683       5685#    5685       5687#    5687       5689#    5689
5691#    5691       5693#    5693       5695#    5695       5697#    5697       5699#    5699       5701#    5701       5703#    5703
5705#    5705       5707#    5707       5709#    5709       5711#    5711       5713#    5713       5715#    5715       5717#    5717
5719#    5719       5721#    5721       5723#    5723       5725#    5725       5727#    5727       5729#    5729       5731#    5731
5733#    5733       5735#    5735       5737#    5737       5739#    5739       5741#    5741       5743#    5743       5745#    5745
5747#    5747       5749#    5749       5752#    5752       5754#    5754       5757#    5757       5759#    5759       5761#    5761
5763#    5763       5765#    5765       5767#    5767       5769#    5769       5771#    5771       5773#    5773       5775#    5775
5777#    5777       5779#    5779       5781#    5781       5783#    5783       5785#    5785       5787#    5787       5789#    5789
5791#    5791       5793#    5793       5795#    5795       5797#    5797       5799#    5799       5801#    5801       5803#    5803
5880#    5880       5882#    5882       5884#    5884       5886#    5886       5888#    5888       5890#    5890       5892#    5892
5894#    5894       5896#    5896       5898#    5898       5901#    5901       5903#    5903       5980#    5980       5982#    5982
5984#    5984       5986#    5986       5988#    5988       6066#    6066       6068#    6068       6070#    6070       6072#    6072
6074#    6074       6076#    6076       6079#    6079       6160#    6160       6162#    6162       6164#    6164       6166#    6166
6168#    6168       6170#    6170       6172#    6172       6174#    6174       6176#    6176       6179#    6179       6181#    6181
6183#    6183       6185#    6185       6188#    6188       6265#    6265       6267#    6267       6269#    6269       6271#    6271
6273#    6273       6275#    6275       6277#    6277       6279#    6279       6281#    6281       6283#    6283       6285#    6285
6287#    6287       6289#    6289       6291#    6291       6293#    6293       6295#    6295       6373#    6373       6375#    6375
6377#    6377       6379#    6379       6455#    6455       6457#    6457       6459#    6459       6461#    6461       6463#    6463
6465#    6465       6467#    6467       6469#    6469       6471#    6471       6473#    6473       6475#    6475       6477#    6477
6479#    6479       6481#    6481       6483#    6483       6485#    6485       6487#    6487       6489#    6489       6491#    6491
6493#    6493       6495#    6495       6497#    6497       6499#    6499       6501#    6501       6503#    6503       6505#    6505
6507#    6507       6509#    6509       6511#    6511       6513#    6513       6515#    6515       6517#    6517       6519#    6519
6521#    6521       6524#    6524       6527#    6527       6602#    6602       6604#    6604       6606#    6606       6608#    6608
6610#    6610       6612#    6612       6614#    6614       6616#    6616       6618#    6618       6620#    6620       6622#    6622
6624#    6624       6626#    6626       6628#    6628       6630#    6630       6632#    6632       6634#    6634       6636#    6636
6638#    6638       6640#    6640       6642#    6642       6644#    6644       6646#    6646       6648#    6648       6650#    6650
6652#    6652       6654#    6654       6656#    6656       6658#    6658       6660#    6660       6662#    6662       6664#    6664
6666#    6666       6669#    6669       6672#    6672       6746#    6746       6748#    6748       6750#    6750       6752#    6752
6754#    6754       6756#    6756       6758#    6758       6760#    6760       6762#    6762       6764#    6764       6766#    6766
6768#    6768       6770#    6770       6772#    6772       6774#    6774       6776#    6776       6778#    6778       6780#    6780
6782#    6782       6784#    6784       6786#    6786       6788#    6788       6790#    6790       6792#    6792       6794#    6794
6796#    6796       6798#    6798       6800#    6800       6802#    6802       6804#    6804       6806#    6806       6808#    6808
```

```
        6810#  6810    6812#  6812    6814#  6814    6816#  6816    6818#  6818    6820#  6820    6822#  6822    SEQ 0890
        6824#  6824    6826#  6826    6828#  6828    6830#  6830    6832#  6832    6834#  6834    6836#  6836
        6838#  6838    6840#  6840    6842#  6842    6845#  6845    6848#  6848    6921#  6921    6923#  6923
        6925#  6925    6927#  6927    6929#  6929    6931#  6931    6933#  6933    6935#  6935    6937#  6937
        6939#  6939    6941#  6941    6943#  6943    6945#  6945    6947#  6947    6949#  6949    6951#  6951
        6953#  6953    6955#  6955    6957#  6957    6959#  6959    6961#  6961    6963#  6963    6965#  6965
        6967#  6967    6969#  6969    6971#  6971    6973#  6973    6975#  6975    6977#  6977    6979#  6979
        6981#  6981    6983#  6983    6985#  6985    6987#  6987    6989#  6989    6991#  6991    6993#  6993
        6995#  6995    6998#  6998    7000#  7000    7002#  7002
%MR      194#   194     196#   196     198#   198     200#   200     298#   298     300#   300     302#   302
         304#   304     306#   306     309#   309     311#   311     313#   313     315#   315     409#   409
         411#   411     413#   413     415#   415     417#   417     419#   419     421#   421     423#   423
         425#   425     427#   427     429#   429     431#   431     433#   433     436#   436     438#   438
         532#   532     534#   534     536#   536     538#   538     540#   540     542#   542     544#   544
         546#   546     548#   548     550#   550     552#   552     554#   554     556#   556     559#   559
         561#   561     665#   665     667#   667     669#   669     671#   671     673#   673     675#   675
         677#   677     679#   679     681#   681     683#   683     685#   685     687#   687     689#   689
         691#   691     693#   693     695#   695     697#   697     699#   699     701#   701     703#   703
         705#   705     707#   707     709#   709     711#   711     797#   797     799#   799     801#   801
         803#   803     888#   888     890#   890     892#   892     894#   894     979#   979     981#   981
         983#   983     985#   985    1067#  1067    1069#  1069    1071#  1071    1149#  1149    1151#  1151
        1153#  1153    1155#  1155    1157#  1157    1159#  1159    1161#  1161    1164#  1164    1240#  1240
        1242#  1242    1244#  1244    1246#  1246    1327#  1327    1329#  1329    1331#  1331    1333#  1333
        1335#  1335    1337#  1337    1339#  1339    1341#  1341    1343#  1343    1345#  1345    1347#  1347
        1349#  1349    1351#  1351    1353#  1353    1356#  1356    1358#  1358    1360#  1360    1363#  1363
        1439#  1439    1441#  1441    1443#  1443    1445#  1445    1447#  1447    1449#  1449    1451#  1451
        1453#  1453    1455#  1455    1457#  1457    1459#  1459    1461#  1461    1463#  1463    1465#  1465
        1467#  1467    1469#  1469    1471#  1471    1473#  1473    1475#  1475    1477#  1477    1479#  1479
        1481#  1481    1483#  1483    1485#  1485    1487#  1487    1489#  1489    1491#  1491    1493#  1493
        1495#  1495    1497#  1497    1499#  1499    1501#  1501    1503#  1503    1505#  1505    1507#  1507
        1509#  1509    1511#  1511    1513#  1513    1515#  1515    1517#  1517    1519#  1519    1521#  1521
        1523#  1523    1526#  1526    1528#  1528    1604#  1604    1606#  1606    1608#  1608    1610#  1610
        1612#  1612    1614#  1614    1616#  1616    1618#  1618    1620#  1620    1622#  1622    1624#  1624
        1626#  1626    1628#  1628    1630#  1630    1632#  1632    1634#  1634    1636#  1636    1638#  1638
        1640#  1640    1642#  1642    1644#  1644    1646#  1646    1648#  1648    1650#  1650    1652#  1652
        1654#  1654    1656#  1656    1658#  1658    1660#  1660    1662#  1662    1664#  1664    1666#  1666
        1668#  1668    1670#  1670    1672#  1672    1674#  1674    1676#  1676    1678#  1678    1680#  1680
        1682#  1682    1684#  1684    1686#  1686    1688#  1688    1691#  1691    1693#  1693    1769#  1769
        1771#  1771    1773#  1773    1775#  1775    1777#  1777    1779#  1779    1781#  1781    1783#  1783
        1785#  1785    1787#  1787    1789#  1789    1791#  1791    1793#  1793    1795#  1795    1797#  1797
        1799#  1799    1801#  1801    1803#  1803    1805#  1805    1807#  1807    1809#  1809    1811#  1811
        1813#  1813    1815#  1815    1817#  1817    1819#  1819    1821#  1821    1823#  1823    1825#  1825
        1827#  1827    1829#  1829    1831#  1831    1833#  1833    1835#  1835    1837#  1837    1839#  1839
        1841#  1841    1843#  1843    1845#  1845    1847#  1847    1849#  1849    1851#  1851    1853#  1853
        1855#  1855    1858#  1858    1860#  1860    1936#  1936    1938#  1938    1940#  1940    1942#  1942
        1944#  1944    1946#  1946    1948#  1948    1950#  1950    1952#  1952    1954#  1954    1956#  1956
        1958#  1958    1960#  1960    1962#  1962    1964#  1964    1966#  1966    1968#  1968    1970#  1970
        1972#  1972    1974#  1974    1976#  1976    1978#  1978    1980#  1980    1982#  1982    1984#  1984
        1986#  1986    1988#  1988    1990#  1990    1992#  1992    1994#  1994    1996#  1996    1998#  1998
        2000#  2000    2002#  2002    2004#  2004    2006#  2006    2008#  2008    2010#  2010    2012#  2012
        2014#  2014    2016#  2016    2018#  2018    2021#  2021    2023#  2023    2099#  2099    2101#  2101
        2103#  2103    2105#  2105    2107#  2107    2109#  2109    2111#  2111    2113#  2113    2115#  2115
        2117#  2117    2119#  2119    2121#  2121    2123#  2123    2125#  2125    2127#  2127    2129#  2129
```

```
2131#   2131    2133#   2133    2135#   2135    2137#   2137    2139#   2139    2141#   2141    2143#   2143
2145#   2145    2147#   2147    2149#   2149    2151#   2151    2153#   2153    2155#   2155    2157#   2157
2159#   2159    2161#   2161    2163#   2163    2165#   2165    2167#   2167    2169#   2169    2171#   2171
2173#   2173    2175#   2175    2177#   2177    2179#   2179    2181#   2181    2183#   2183    2185#   2185
2188#   2188    2190#   2190    2269#   2269    2271#   2271    2273#   2273    2275#   2275    2277#   2277
2279#   2279    2281#   2281    2361#   2361    2363#   2363    2365#   2365    2367#   2367    2369#   2369
2371#   2371    2373#   2373    2375#   2375    2454#   2454    2456#   2456    2458#   2458    2460#   2460
2462#   2462    2464#   2464    2466#   2466    2468#   2468    2470#   2470    2549#   2549    2551#   2551
2553#   2553    2555#   2555    2557#   2557    2559#   2559    2561#   2561    2563#   2563    2565#   2565
2567#   2567    2646#   2646    2648#   2648    2650#   2650    2652#   2652    2654#   2654    2656#   2656
2658#   2658    2660#   2660    2662#   2662    2664#   2664    2666#   2666    2752#   2752    2754#   2754
2756#   2756    2758#   2758    2760#   2760    2762#   2762    2764#   2764    2766#   2766    2769#   2769
2771#   2771    2773#   2773    2775#   2775    2854#   2854    2856#   2856    2858#   2858    2860#   2860
2862#   2862    2864#   2864    2866#   2866    2869#   2869    2871#   2871    2873#   2873    2875#   2875
2877#   2877    2879#   2879    2881#   2881    2884#   2884    2886#   2886    2888#   2888    2890#   2890
2892#   2892    2894#   2894    2896#   2896    2899#   2899    2977#   2977    2979#   2979    2981#   2981
2983#   2983    2985#   2985    2987#   2987    2989#   2989    3064#   3064    3066#   3066    3152#   3152
3154#   3154    3156#   3156    3158#   3158    3160#   3160    3162#   3162    3164#   3164    3166#   3166
3168#   3168    3170#   3170    3252#   3252    3254#   3254    3256#   3256    3258#   3258    3260#   3260
3263#   3263    3265#   3265    3267#   3267    3269#   3269    3272#   3272    3274#   3274    3276#   3276
3279#   3279    3281#   3281    3283#   3283    3286#   3286    3288#   3288    3290#   3290    3293#   3293
3295#   3295    3297#   3297    3300#   3300    3302#   3302    3304#   3304    3307#   3307    3309#   3309
3311#   3311    3314#   3314    3316#   3316    3318#   3318    3321#   3321    3323#   3323    3325#   3325
3328#   3328    3330#   3330    3332#   3332    3335#   3335    3337#   3337    3339#   3339    3342#   3342
3344#   3344    3346#   3346    3349#   3349    3351#   3351    3353#   3353    3356#   3356    3358#   3358
3360#   3360    3363#   3363    3365#   3365    3367#   3367    3370#   3370    3372#   3372    3374#   3374
3377#   3377    3379#   3379    3381#   3381    3384#   3384    3386#   3386    3388#   3388    3391#   3391
3393#   3393    3395#   3395    3398#   3398    3400#   3400    3480#   3480    3482#   3482    3484#   3484
3486#   3486    3488#   3488    3490#   3490    3492#   3492    3494#   3494    3496#   3496    3498#   3498
3500#   3500    3503#   3503    3505#   3505    3589#   3589    3591#   3591    3593#   3593    3595#   3595
3598#   3598    3600#   3600    3602#   3602    3605#   3605    3607#   3607    3609#   3609    3612#   3612
3614#   3614    3616#   3616    3619#   3619    3621#   3621    3702#   3702    3705#   3705    3707#   3707
3709#   3709    3712#   3712    3714#   3714    3716#   3716    3719#   3719    3721#   3721    3723#   3723
3726#   3726    3728#   3728    3730#   3730    3733#   3733    3735#   3735    3738#   3738    3814#   3814
3816#   3816    3818#   3818    3820#   3820    3822#   3822    3824#   3824    3826#   3826    3828#   3828
3920#   3920    3922#   3922    3924#   3924    3926#   3926    3928#   3928    3931#   3931    3933#   3933
3935#   3935    3937#   3937    3939#   3939    3941#   3941    3943#   3943    3945#   3945    3948#   3948
4030#   4030    4032#   4032    4034#   4034    4036#   4036    4038#   4038    4040#   4040    4042#   4042
4044#   4044    4046#   4046    4048#   4048    4050#   4050    4052#   4052    4055#   4055    4136#   4136
4138#   4138    4140#   4140    4142#   4142    4144#   4144    4147#   4147    4149#   4149    4151#   4151
4238#   4238    4240#   4240    4242#   4242    4244#   4244    4246#   4246    4251#   4251    4253#   4253
4255#   4255    4257#   4257    4259#   4259    4262#   4262    4343#   4343    4345#   4345    4347#   4347
4349#   4349    4351#   4351    4353#   4353    4355#   4355    4357#   4357    4359#   4359    4361#   4361
4363#   4363    4366#   4366    4368#   4368    4370#   4370    4449#   4449    4451#   4451    4453#   4453
4455#   4455    4457#   4457    4459#   4459    4461#   4461    4463#   4463    4465#   4465    4467#   4467
4470#   4470    4473#   4473    4552#   4552    4555#   4555    4557#   4557    4559#   4559    4561#   4561
4563#   4563    4565#   4565    4567#   4567    4569#   4569    4572#   4572    4574#   4574    4576#   4576
4578#   4578    4580#   4580    4582#   4582    4584#   4584    4586#   4586    4589#   4589    4591#   4591
4593#   4593    4595#   4595    4597#   4597    4599#   4599    4601#   4601    4603#   4603    4606#   4606
4608#   4608    4610#   4610    4612#   4612    4614#   4614    4616#   4616    4618#   4618    4620#   4620
4623#   4623    4625#   4625    4627#   4627    4629#   4629    4631#   4631    4633#   4633    4635#   4635
4637#   4637    4640#   4640    4642#   4642    4644#   4644    4646#   4646    4648#   4648    4650#   4650
4652#   4652    4654#   4654    4657#   4657    4659#   4659    4661#   4661    4663#   4663    4665#   4665
```

```
4667#  4667   4669#  4669   4671#  4671   4674#  4674   4676#  4676   4678#  4678   4680#  4680
4682#  4682   4684#  4684   4686#  4686   4688#  4688   4691#  4691   4693#  4693   4695#  4695
4697#  4697   4699#  4699   4701#  4701   4703#  4703   4705#  4705   4708#  4708   4710#  4710
4712#  4712   4714#  4714   4716#  4716   4718#  4718   4720#  4720   4722#  4722   4725#  4725
4727#  4727   4729#  4729   4731#  4731   4733#  4733   4735#  4735   4737#  4737   4739#  4739
4742#  4742   4744#  4744   4746#  4746   4748#  4748   4750#  4750   4752#  4752   4754#  4754
4756#  4756   4759#  4759   4761#  4761   4763#  4763   4765#  4765   4767#  4767   4769#  4769
4771#  4771   4773#  4773   4775#  4775   4853#  4853   4855#  4855   4857#  4857   4859#  4859
4861#  4861   4863#  4863   4865#  4865   4867#  4867   4869#  4869   4871#  4871   4873#  4873
4875#  4875   4970#  4970   4972#  4972   4974#  4974   4976#  4976   4978#  4978   4980#  4980
4982#  4982   4984#  4984   4986#  4986   4989#  4989   4991#  4991   4993#  4993   4995#  4995
4997#  4997   4999#  4999   5001#  5001   5003#  5003   5005#  5005   5008#  5008   5010#  5010
5012#  5012   5014#  5014   5016#  5016   5018#  5018   5020#  5020   5022#  5022   5024#  5024
5027#  5027   5029#  5029   5031#  5031   5033#  5033   5035#  5035   5037#  5037   5039#  5039
5041#  5041   5043#  5043   5046#  5046   5048#  5048   5050#  5050   5052#  5052   5054#  5054
5056#  5056   5058#  5058   5060#  5060   5063#  5063   5174#  5174   5177#  5177   5180#  5180
5183#  5183   5194#  5194   5196#  5196   5199#  5199   5201#  5201   5203#  5203   5206#  5206
5208#  5208   5210#  5210   5213#  5213   5215#  5215   5217#  5217   5220#  5220   5222#  5222
5224#  5224   5227#  5227   5229#  5229   5231#  5231   5234#  5234   5236#  5236   5238#  5238
5241#  5241   5243#  5243   5245#  5245   5248#  5248   5250#  5250   5252#  5252   5255#  5255
5257#  5257   5259#  5259   5262#  5262   5264#  5264   5266#  5266   5269#  5269   5271#  5271
5273#  5273   5276#  5276   5278#  5278   5280#  5280   5283#  5283   5285#  5285   5287#  5287
5290#  5290   5292#  5292   5294#  5294   5297#  5297   5299#  5299   5301#  5301   5304#  5304
5306#  5306   5308#  5308   5311#  5311   5313#  5313   5315#  5315   5318#  5318   5320#  5320
5322#  5322   5325#  5325   5327#  5327   5329#  5329   5332#  5332   5334#  5334   5336#  5336
5339#  5339   5341#  5341   5343#  5343   5346#  5346   5348#  5348   5350#  5350   5353#  5353
5355#  5355   5357#  5357   5360#  5360   5362#  5362   5364#  5364   5367#  5367   5369#  5369
5371#  5371   5374#  5374   5376#  5376   5378#  5378   5381#  5381   5383#  5383   5386#  5386
5388#  5388   5390#  5390   5392#  5392   5394#  5394   5396#  5396   5398#  5398   5400#  5400
5402#  5402   5404#  5404   5406#  5406   5408#  5408   5410#  5410   5412#  5412   5414#  5414
5416#  5416   5418#  5418   5420#  5420   5422#  5422   5424#  5424   5426#  5426   5428#  5428
5430#  5430   5432#  5432   5434#  5434   5436#  5436   5521#  5521   5523#  5523   5525#  5525
5527#  5527   5529#  5529   5531#  5531   5533#  5533   5535#  5535   5537#  5537   5539#  5539
5542#  5542   5544#  5544   5546#  5546   5548#  5548   5550#  5550   5552#  5552   5554#  5554
5634#  5634   5636#  5636   5638#  5638   5640#  5640   5642#  5642   5644#  5644   5646#  5646
5648#  5648   5650#  5650   5652#  5652   5655#  5655   5657#  5657   5659#  5659   5661#  5661
5663#  5663   5665#  5665   5667#  5667   5669#  5669   5671#  5671   5673#  5673   5675#  5675
5677#  5677   5679#  5679   5681#  5681   5683#  5683   5685#  5685   5687#  5687   5689#  5689
5691#  5691   5693#  5693   5695#  5695   5697#  5697   5699#  5699   5701#  5701   5703#  5703
5705#  5705   5707#  5707   5709#  5709   5711#  5711   5713#  5713   5715#  5715   5717#  5717
5719#  5719   5721#  5721   5723#  5723   5725#  5725   5727#  5727   5729#  5729   5731#  5731
5733#  5733   5735#  5735   5737#  5737   5739#  5739   5741#  5741   5743#  5743   5745#  5745
5747#  5747   5749#  5749   5752#  5752   5754#  5754   5757#  5757   5759#  5759   5761#  5761
5763#  5763   5765#  5765   5767#  5767   5769#  5769   5771#  5771   5773#  5773   5775#  5775
5777#  5777   5779#  5779   5781#  5781   5783#  5783   5785#  5785   5787#  5787   5789#  5789
5791#  5791   5793#  5793   5795#  5795   5797#  5797   5799#  5799   5801#  5801   5803#  5803
5880#  5880   5882#  5882   5884#  5884   5886#  5886   5888#  5888   5890#  5890   5892#  5692
5894#  5894   5896#  5896   5898#  5898   5901#  5901   5903#  5903   5980#  5980   5982#  5982
5984#  5984   5986#  5986   5988#  5988   6066#  6066   6068#  6068   6070#  6070   6072#  6072
6074#  6074   6076#  6076   6079#  6079   6160#  6160   6162#  6162   6164#  6164   6166#  6166
6168#  6168   6170#  6170   6172#  6172   6174#  6174   6176#  6176   6179#  6179   6181#  6181
6183#  6183   6185#  6185   6188#  6188   6265#  6265   6267#  6267   6269#  6269   6271#  6271
6273#  6273   6275#  6275   6277#  6277   6279#  6279   6281#  6281   6283#  6283   6285#  6285
```

```
0880         6287#  6287   6289#  6289   6291#  6291   6293#  6293   6295#  6295   6373#  6373   6375#  6375     SEQ 0893
             6377#  6377   6379#  6379   6455#  6455   6457#  6457   6459#  6459   6461#  6461   6463#  6463
             6465#  6465   6467#  6467   6469#  6469   6471#  6471   6473#  6473   6475#  6475   6477#  6477
             6479#  6479   6481#  6481   6483#  6483   6485#  6485   6487#  6487   6489#  6489   6491#  6491
             6493#  6493   6495#  6495   6497#  6497   6499#  6499   6501#  6501   6503#  6503   6505#  6505
             6507#  6507   6509#  6509   6511#  6511   6513#  6513   6515#  6515   6517#  6517   6519#  6519
             6521#  6521   6524#  6524   6527#  6527   6602#  6602   6604#  6604   6606#  6606   6608#  6608
             6610#  6610   6612#  6612   6614#  6614   6616#  6616   6618#  6618   6620#  6620   6622#  6622
             6624#  6624   6626#  6626   6628#  6628   6630#  6630   6632#  6632   6634#  6634   6636#  6636
             6638#  6638   6640#  6640   6642#  6642   6644#  6644   6646#  6646   6648#  6648   6650#  6650
             6652#  6652   6654#  6654   6656#  6656   6658#  6658   6660#  6660   6662#  6662   6664#  6664
             6666#  6666   6669#  6669   6672#  6672   6746#  6746   6748#  6748   6750#  6750   6752#  6752
             6754#  6754   6756#  6756   6758#  6758   6760#  6760   6762#  6762   6764#  6764   6766#  6766
             6768#  6768   6770#  6770   6772#  6772   6774#  6774   6776#  6776   6778#  6778   6780#  6780
             6782#  6782   6784#  6784   6786#  6786   6788#  6788   6790#  6790   6792#  6792   6794#  6794
             6796#  6796   6798#  6798   6800#  6800   6802#  6802   6804#  6804   6806#  6806   6808#  6808
             6810#  6810   6812#  6812   6814#  6814   6816#  6816   6818#  6818   6820#  6820   6822#  6822
             6824#  6824   6826#  6826   6828#  6828   6830#  6830   6832#  6832   6834#  6834   6836#  6836
             6838#  6838   6840#  6840   6842#  6842   6845#  6845   6848#  6848   6921#  6921   6923#  6923
             6925#  6925   6927#  6927   6929#  6929   6931#  6931   6933#  6933   6935#  6935   6937#  6937
             6939#  6939   6941#  6941   6943#  6943   6945#  6945   6947#  6947   6949#  6949   6951#  6951
             6953#  6953   6955#  6955   6957#  6957   6959#  6959   6961#  6961   6963#  6963   6965#  6965
             6967#  6967   6969#  6969   6971#  6971   6973#  6973   6975#  6975   6977#  6977   6979#  6979
             6981#  6981   6983#  6983   6985#  6985   6987#  6987   6989#  6989   6991#  6991   6993#  6993
             6995#  6995   6998#  6998   7000#  7000   7002#  7002
.LADDR        194    298    309    311    313    315    409    411    436    438    532    534    559    561
              665    675    679    683    687    691    695    699    703    707    711    797    799    801
              803    888    890    892    894    979    981    983    985   1067   1069   1071   1149   1164
             1240   1246   1327   1356   1363   1439   1449   1451   1453   1455   1457   1459   1461   1463
             1465   1467   1469   1471   1473   1475   1477   1479   1481   1483   1485   1487   1489   1491
             1493   1495   1497   1499   1501   1503   1505   1507   1509   1511   1513   1515   1517   1519
             1521   1523   1526   1528   1604   1614   1616   1618   1620   1622   1624   1626   1628   1630
             1632   1634   1636   1638   1640   1642   1644   1646   1648   1650   1652   1654   1656   1658
             1660   1662   1664   1666   1668   1670   1672   1674   1676   1678   1680   1682   1684   1686
             1688   1691   1693   1769   1781   1783   1785   1787   1789   1791   1793   1795   1797   1799
             1801   1803   1805   1807   1809   1811   1813   1815   1817   1819   1821   1823   1825   1827
             1829   1831   1833   1835   1837   1839   1841   1843   1845   1847   1849   1851   1853   1855
             1858   1860   1936   1948   1950   1952   1954   1956   1958   1960   1962   1964   1966   1968
             1970   1972   1974   1976   1978   1980   1982   1984   1986   1988   1990   1992   1994   1996
             1998   2000   2002   2004   2006   2008   2010   2012   2014   2016   2018   2021   2023   2099
             2111   2113   2115   2117   2119   2121   2123   2125   2127   2129   2131   2133   2135   2137
             2139   2141   2143   2145   2147   2149   2151   2153   2155   2157   2159   2161   2163   2165
             2167   2169   2171   2173   2175   2177   2179   2181   2183   2185   2188   2190   2269   2273
             2275   2277   2279   2281   2361   2365   2367   2369   2371   2373   2375   2454   2458   2460
             2462   2464   2466   2468   2470   2549   2553   2555   2557   2559   2561   2563   2565   2567
             2646   2650   2652   2654   2656   2658   2660   2662   2664   2666   2752   2769   2771   2773
             2775   2854   2856   2858   2860   2862   2864   2866   2869   2871   2873   2875   2877   2879
             2881   2884   2886   2888   2890   2892   2894   2896   2899   2977   2979   2981   2983   2985
             2987   2989   3064   3152   3154   3160   3166   3252   3254   3260   3263   3269   3272   3279
             3286   3293   3300   3307   3314   3321   3328   3335   3342   3349   3356   3363   3370   3377
             3384   3391   3480   3503   3505   3589   3591   3702   3735   3738   3814   3815   3920   3948
             4030   4032   4055   4136   4138   4238   4240   4251   4253   4343   4366   4449   4470   4473
             4552   4555   4759   4761   4853   4873   4970   4989   5008   5027   5046   5063   5174   5177
```

```
          5180  5183  5194  5196  5199  5386  5388  5390  5392  5394  5396  5398  5400  5402
          5404  5406  5408  5410  5412  5414  5416  5418  5420  5422  5424  5426  5428  5430
          5432  5434  5436  5521  5523  5554  5634  5636  5655  5757  5759  5761  5763  5765
          5767  5769  5771  5773  5775  5777  5779  5781  5783  5785  5787  5789  5791  5793
          5795  5797  5799  5801  5803  5880  5882  5901  5903  5980  5982  6066  6072  6079
          6160  6188  6265  6295  6373  6379  6455  6457  6527  6602  6604  6672  6746  6748
          6848  6921  6923  6998  7002
.LCCOF     194   198   309   311   409   415   427   532   538   550   797   803   888   894
           979   985  1159  1246  1356  2752  2764  2860  2985  3156  3162  3168  3256  3265
          3274  3281  3288  3295  3302  3309  3316  3323  3330  3337  3344  3351  3358  3365
          3372  3379  3386  3393  3482  3494  3498  3593  3614  3707  3728  3818  3824  3931
          3939  4034  4036  4040  4042  4046  4048  4140  4142  4149  4151  4242  4244  4246
          4255  4257  4259  4355  4361  4459  4465  4761  4769  4863  4867  5046  5054  5174
          5180  5203  5210  5217  5224  5231  5238  5245  5252  5259  5266  5273  5280  5287
          5294  5301  5308  5315  5322  5329  5336  5343  5350  5357  5364  5371  5378  5525
          5533  5544  5546  5548  5550  5552  5636  5644  5652  5884  5892  5898  5984  6070
          6074  6164  6179  6277  6281  6377  6459  6461  6465  6469  6473  6477  6481  6485
          6489  6493  6497  6501  6505  6509  6511  6517  6519  6521  6606  6608  6612  6616
          6620  6624  6628  6632  6636  6640  6644  6648  6652  6656  6662  6664  6666  6750
          6754  6758  6762  6766  6770  6774  6776  6782  6786  6790  6796  6800  6804  6808
          6812  6814  6818  6822  6828  6832  6836  6840  6845  6925  6929  6933  6937  6941
          6945  6949  6951  6957  6961  6965  6971  6973  6977  6981  6987  6991  6995
.LCCON     196   200   313   315   421   433   544   556   799   801   890   892   981   983
          1358  1360  2756  2760  2875  2890  3486  3490  3600  3607  3714  3721  3820  3822
          3935  3943  4357  4359  4765  4773  4865  4869  5050  5058  5177  5183  5529  5537
          5640  5648  5886  5888  5894  5896  6181  6183  6275  6283  6457  6463  6467  6471
          6475  6479  6483  6487  6491  6495  6499  6503  6507  6513  6515  6524  6604  6610
          6614  6618  6622  6626  6630  6634  6638  6642  6646  6650  6654  6658  6660  6669
          6752  6756  6760  6764  6768  6772  6778  6780  6784  6788  6792  6794  6798  6802
          6806  6810  6816  6820  6824  6826  6830  6834  6838  6842  6927  6931  6935  6939
          6943  6947  6953  6955  6959  6963  6967  6969  6975  6979  6983  6985  6989  6993
.LCENA     801   803   892   894   983   985
.LCJP      797   799   801   803  6469  6471  6473  6475  6616  6618  6620  6622  6766  6768
          6770  6772  6941  6943  6945  6947
.LCJPP    2752  2756  2760  2764  2860  2875  2890  2985  4970  4972  4974  4976  4978  4989
          4991  4993  4995  4997  5008  5010  5012  5014  5016  5027  5029  5031  5033  5035
          6521  6666  6814  6816  6818  6820  6973  6975  6977  6979
.LCJS      979   981   983   985  1067  1155  1159  1240  1329  1333  1337  1341  1345  1349
          1439  1441  1443  1445  1449  1453  1457  1461  1465  1469  1473  1477  1481  1485
          1489  1493  1497  1501  1505  1509  1513  1515  1517  1519  1604  1608  1610  1614
          1618  1622  1626  1630  1634  1638  1642  1646  1650  1654  1658  1662  1666  1670
          1674  1678  1680  1682  1684  1769  1773  1777  1781  1785  1789  1793  1797  1801
          1805  1809  1813  1817  1821  1825  1829  1833  1837  1841  1845  1847  1849  1851
          1936  1940  1944  1948  1952  1956  1960  1964  1968  1972  1976  1980  1984  1988
          1992  1996  2000  2004  2008  2012  2014  2016  2018  2099  2103  2107  2111  2115
          2119  2123  2127  2131  2135  2139  2143  2147  2151  2155  2159  2163  2167  2171
          2175  2177  2179  2181  2269  2273  2275  2277  2279  2361  2365  2367  2369  2371
          2454  2458  2460  2462  2464  2549  2553  2555  2557  2559  2646  2650  2652  2654
          2656  2854  2856  2869  2871  2884  2886  2977  2981  3920  3922  3924  3926  4557
          4561  4565  4569  4574  4578  4582  4586  4591  4595  4599  4603  4608  4612  4616
          4620  4625  4629  4633  4637  4642  4646  4650  4654  4659  4663  4667  4671  4676
          4680  4684  4688  4693  4697  4701  4705  4710  4714  4718  4722  4727  4731  4735
          4739  4744  4748  4752  4756  4984  5003  5022  5041  5201  5208  5215  5222  5229
          5236  5243  5250  5257  5264  5271  5278  5285  5292  5299  5306  5313  5320  5327
          5334  5341  5348  5355  5362  5369  5376  6068  6162  6166  6170  6174  6517  6662
          6758  6760  6762  6764  6933  6935  6937  6939
.LCJV      888   890   892   894  6477  6479  6481  6483  6624  6626  6628  6630  6782  6784
          6786  6788  6957  6959  6961  6963
.LCONT     667   669   671   673   677   681   685   689   693   697   701   705   709  1149
          1151  1153  3398  3619  4052  6493  6495  6497  6499  6640  6642  6644  6646  6828
          6830  6832  6834  6987  6989  6991  6993
.LCRTN    1071  1161  1246  1356  1358  1360  1528  1693  1860  2023  2190  2281  2373  2375
          2466  2468  2470  2561  2563  2565  2567  2658  2660  2662  2664  2666  2864  2879
          2894  2987  3709  3716  3723  3730  3733  3933  3937  3941  3945  4370  4470  4871
          4986  5005  5024  5043  6072  6185  6285  6519  6524  6664  6669  6812  6971
.LDISA     194   196   309   313   415   421   538   544   797   799   888   890   979   981
          1356  1358  2752  2756  2860  2875  3482  3486  3593  3600  3707  3714  3818  3820
          3931  3935  4034  4040  4046  4355  4357  4761  4765  4863  4865  5046  5050  5174
          5177  5525  5529  5544  5546  5548  5550  5636  5640  5884  5886  5892  5894  6179
          6181  6281  6283  6461  6463  6469  6471  6477  6479  6485  6487  6493  6495  6501
          6503  6511  6513  6608  6610  6616  6618  6624  6626  6632  6634  6640  6642  6648
          6650  6656  6658  6750  6752  6758  6760  6766  6768  6776  6778  6782  6784  6790
          6792  6796  6798  6804  6806  6814  6816  6822  6824  6828  6830  6836  6838  6925
          6927  6933  6935  6941  6943  6951  6953  6957  6959  6965  6967  6973  6975  6981
          6983  6987  6989
.LENA      198   200   311   315   409   427   433   532   550   556  1159  1246  1360  2760
          2764  2890  2985  3156  3162  3168  3256  3265  3274  3281  3288  3295  3302  3309
          3316  3323  3330  3337  3344  3351  3358  3365  3372  3379  3386  3393  3490  3494
          3498  3607  3614  3721  3728  3822  3824  3939  3943  4036  4042  4048  4140  4142
          4149  4151  4242  4244  4246  4255  4257  4259  4359  4361  4459  4465  4769  4773
          4867  4869  5054  5058  5180  5183  5203  5210  5217  5224  5231  5238  5245  5252
          5259  5266  5273  5280  5287  5294  5301  5308  5315  5322  5329  5336  5343  5350
          5357  5364  5371  5378  5533  5537  5552  5644  5648  5652  5888  5896  5898  5984
          6070  6074  6164  6183  6275  6277  6377  6457  6459  6465  6467  6473  6475  6481
          6483  6489  6491  6497  6499  6505  6507  6509  6515  6517  6519  6521  6524  6604
          6606  6612  6614  6620  6622  6628  6630  6636  6638  6644  6646  6652  6654  6660
          6662  6664  6666  6669  6754  6756  6762  6764  6770  6772  6774  6780  6786  6788
          6794  6800  6802  6808  6810  6812  6818  6820  6826  6832  6834  6840  6842  6845
          6929  6931  6937  6939  6945  6947  6949  6955  6961  6963  6969  6971  6977  6979
          6985  6991  6993  6995
.LJ        194   196   198   200   300   302   304   306   309   311   313   315   409   411
           413   415   417   419   421   423   425   427   429   431   433   436   438   532
           534   536   538   540   542   544   546   548   550   552   554   556   559   561
           665   667   669   671   673   675   677   679   681   683   685   687   689   691
           693   695   697   699   701   703   705   707   709   711   797   799   801   803
           888   890   892   894   979   981   983   985  1067  1155  1159  1164  1240  1242
          1327  1329  1331  1333  1335  1337  1339  1341  1343  1345  1347  1349  1351  1363
          1439  1441  1443  1445  1447  1449  1451  1453  1455  1457  1459  1461  1463  1465
          1467  1469  1471  1473  1475  1477  1479  1481  1483  1485  1487  1489  1491  1493
          1495  1497  1499  1501  1503  1505  1507  1509  1511  1513  1515  1517  1519  1521
          1523  1604  1606  1608  1610  1612  1614  1616  1618  1620  1622  1624  1626  1628
          1630  1632  1634  1636  1638  1640  1642  1644  1646  1648  1650  1652  1654  1656
          1658  1660  1662  1664  1666  1668  1670  1672  1674  1676  1678  1680  1682  1684
          1686  1688  1769  1771  1773  1775  1777  1779  1781  1783  1785  1787  1789  1791
          1793  1795  1797  1799  1801  1803  1805  1807  1809  1811  1813  1815  1817  1819
          1821  1823  1825  1827  1829  1831  1833  1835  1837  1839  1841  1843  1845  1847
```

SEQ 0883

1849
1956
1984
2012
2119
2147
2175
2369
2652
2775
2886
3160
3269
3302
3335
3367
3480
3595
3714
3922
4044
4242
4363
4569
4599
4629
4659
4688
4718
4748
4861
4989
5020
5054
5206
5238
5271
5304
5336
5369
5400
5428
5542
5648
5677
5705
5733
5763
5791
5896
6168
6277
6465

1851
1958
1986
2014
2121
2149
2177
2371
2654
2854
2888
3162
3272
3304
3337
3370
3482
3598
3719
3924
4046
4244
4366
4572
4601
4631
4661
4691
4720
4750
4863
4991
5022
5056
5208
5241
5273
5306
5339
5371
5402
5430
5544
5650
5679
5707
5735
5765
5793
5901
6170
6279
6467

1853
1960
1988
2016
2123
2151
2179
2454
2656
2856
2890
3164
3274
3307
3339
3372
3484
3600
3721
3926
4048
4246
4457
4574
4603
4633
4663
4693
4722
4752
4865
4993
5027
5058
5210
5243
5276
5308
5341
5374
5404
5432
5546
5652
5681
5709
5737
5767
5795
5903
6172
6281
6469

1855
1962
1990
2018
2125
2153
2181
2458
2752
2858
2892
3166
3276
3309
3342
3374
3486
3602
3726
3931
4050
4251
4461
4576
4606
4635
4665
4695
4725
4754
4867
4995
5029
5060
5213
5245
5278
5311
5343
5376
5406
5434
5548
5655
5683
5711
5739
5769
5797
5982
6174
6283
6471

1936
1964
1992
2099
2127
2155
2183
2460
2754
2860
2896
3168
3279
3311
3344
3377
3488
3605
3728
3935
4055
4253
4463
4578
4608
4637
4667
4697
4727
4756
4869
4997
5031
5063
5215
5248
5280
5313
5346
5378
5408
5436
5550
5657
5685
5713
5741
5771
5799
5984
6176
6287
6473

1938
1966
1994
2101
2129
2157
2185
2462
2756
2862
2977
3170
3281
3314
3346
3379
3490
3607
3735
3939
4138
4255
4465
4580
4610
4640
4669
4699
4729
4759
4873
4999
5033
5174
5217
5250
5283
5315
5348
5381
5410
5523
5552
5659
5687
5715
5743
5773
5801
5986
6179
6289
6475

1940
1968
1996
2103
2131
2159
2269
2464
2758
2866
2981
3252
3283
3316
3349
3381
3492
3609
3816
3943
4140
4257
4467
4582
4612
4642
4671
4701
4731
4761
4970
5001
5035
5177
5220
5252
5285
5318
5350
5383
5412
5525
5554
5661
5689
5717
5745
5775
5803
5988
6181
6291
6477

1942
1970
1998
2105
2133
2161
2273
2549
2760
2869
2985
3254
3286
3318
3351
3384
3494
3612
3818
3948
4142
4259
4555
4584
4614
4644
4674
4703
4733
4763
4972
5003
5037
5180
5222
5255
5287
5320
5353
5386
5414
5527
5634
5663
5691
5719
5747
5777
5882
6066
6183
6293
6479

1944
1972
2000
2107
2135
2163
2275
2553
2762
2871
2989
3256
3288
3321
3353
3386
3496
3614
3820
4032
4144
4262
4557
4586
4616
4646
4676
4705
4735
4765
4974
5008
5039
5183
5224
5257
5290
5322
5355
5388
5416
5529
5636
5665
5693
5721
5749
5779
5884
6068
6265
6373
6481

1946
1974
2002
2109
2137
2165
2277
2555
2764
2873
3064
3258
3290
3323
3356
3388
3498
3616
3822
4034
4147
4353
4559
4589
4618
4648
4678
4708
4737
4767
4976
5010
5041
5194
5227
5259
5292
5325
5357
5390
5418
5531
5638
5667
5695
5723
5752
5781
5886
6074
6267
6375
6483

1948
1976
2004
2111
2139
2167
2279
2557
2766
2875
3066
3260
3293
3325
3358
3391
3500
3621
3824
4036
4149
4355
4561
4591
4620
4650
4680
4710
4739
4769
4978
5012
5046
5196
5229
5262
5294
5327
5360
5392
5420
5533
5640
5669
5697
5725
5754
5783
5888
6076
6269
6377
6485

1950
1978
2006
2113
2141
2169
2361
2559
2769
2877
3154
3263
3295
3328
3360
3393
3503
3705
3826
4038
4151
4357
4563
4593
4623
4652
4682
4712
4742
4771
4980
5014
5048
5199
5231
5264
5297
5329
5362
5394
5422
5535
5642
5671
5699
5727
5757
5785
5890
6160
6271
6457
6487

1952
1980
2008
2115
2143
2171
2365
2646
2771
2881
3156
3265
3297
3330
3363
3395
3591
3707
3828
4040
4238
4359
4565
4595
4625
4654
4684
4714
4744
4773
4982
5016
5050
5201
5234
5266
5299
5332
5364
5396
5424
5537
5644
5673
5701
5729
5759
5787
5892
6162
6273
6461
6489

1954
1982
2010
2117
2145
2173
2367
2650
2773
2884
3158
3267
3300
3332
3365
3400
3593
3712
3920
4042
4240
4361
4567
4597
4627
4657
4686
4716
4746
4775
4984
5018
5052
5203
5236
5269
5301
5334
5367
5398
5426
5539
5646
5675
5703
5731
5761
5789
5894
6166
6275
6463
6491

SEQ 0896

```
          6493  6495  6497  6499  6501  6503  6505  6507  6509  6511  6513  6515  6517  6519
          6521  6527  6604  6608  6610  6612  6614  6616  6618  6620  6622  6624  6626  6628
          6630  6632  6634  6636  6638  6640  6642  6644  6646  6648  6650  6652  6654  6656
          6658  6660  6662  6664  6666  6672  6748  6750  6752  6754  6756  6758  6760  6762
          6764  6766  6768  6770  6772  6774  6776  6778  6780  6782  6784  6786  6788  6790
          6792  6794  6796  6798  6800  6802  6804  6806  6808  6810  6812  6814  6816  6818
          6820  6822  6824  6826  6828  6830  6832  6834  6836  6838  6840  6842  6845  6848
          6923  6925  6927  6929  6931  6933  6935  6937  6939  6941  6943  6945  6947  6949
          6951  6953  6955  6957  6959  6961  6963  6965  6967  6969  6971  6973  6975  6977
          6979  6981  6983  6985  6987  6989  6991  6993  6995  6998  7000  7002
.LJMAP     194   196   198   200   300   302   304   306   411   417   423   429   436   438
           534   540   546   552   559   561   665   675   679   683   687   691   695   699
           703   707   711  1164  1242  1351  1363  1447  1451  1455  1459  1463  1467  1471
          1475  1479  1483  1487  1491  1495  1499  1503  1507  1511  1521  1523  1606  1612
          1616  1620  1624  1628  1632  1636  1640  1644  1648  1652  1656  1660  1664  1668
          1672  1676  1686  1688  1771  1775  1779  1783  1787  1791  1795  1799  1803  1807
          1811  1815  1819  1823  1827  1831  1835  1839  1843  1853  1855  1938  1942  1946
          1950  1954  1958  1962  1966  1970  1974  1978  1982  1986  1990  1994  1998  2002
          2006  2010  2101  2105  2109  2113  2117  2121  2125  2129  2133  2137  2141  2145
          2149  2153  2157  2161  2165  2169  2173  2183  2185  2754  2758  2762  2766  2769
          2771  2773  2775  2858  2862  2866  2873  2877  2881  2888  2892  2896  2989  3064
          3158  3164  3170  3252  3258  3260  3267  3269  3276  3283  3290  3297  3304  3311
          3318  3325  3332  3339  3346  3353  3360  3367  3374  3381  3388  3395  3400  3484
          3488  3492  3496  3500  3503  3595  3602  3609  3616  3621  3735  3826  3828  3948
          4038  4044  4050  4055  4144  4238  4251  4262  4363  4366  4463  4467  4759  4763
          4767  4771  4775  4873  5048  5052  5056  5060  5063  5194  5196  5381  5383  5386
          5388  5390  5392  5394  5396  5398  5400  5402  5404  5406  5408  5410  5412  5414
          5416  5418  5420  5422  5424  5426  5428  5430  5432  5434  5436  5527  5531  5535
          5539  5554  5634  5638  5642  5646  5650  5657  5661  5665  5669  5673  5677  5681
          5685  5689  5693  5697  5701  5705  5709  5713  5717  5721  5725  5729  5733  5737
          5741  5745  5749  5752  5754  5757  5759  5761  5763  5765  5767  5769  5771  5773
          5775  5777  5779  5781  5783  5785  5787  5789  5791  5793  5795  5797  5799  5801
          5803  5901  5903  5986  5988  6076  6168  6172  6176  6279  6287  6289  6291  6293
          6461  6463  6465  6467  6527  6608  6610  6612  6614  6672  6750  6752  6754  6756
          6848  6925  6927  6929  6931  6998  7000  7002
.LJRP      409   532  3156  3162  3168  3256  3265  3274  3281  3288  3295  3302  3309  3316
          3323  3330  3337  3344  3351  3358  3365  3372  3379  3386  3393  3482  3486  3490
          3494  3498  4036  4042  4048  4142  4151  4361  5203  5210  5217  5224  5231  5238
          5245  5252  5259  5266  5273  5280  5287  5294  5301  5308  5315  5322  5329  5336
          5343  5350  5357  5364  5371  5378  5552  5652  5898  6070  6164  6509  6511  6513
          6515  6656  6658  6660  6790  6792  6794  6845  6965  6967  6969  6995
.LJSRP    3593  3600  3607  3614  3707  3714  3721  3728  3818  3820  3822  3824  6776  6778
          6780  6951  6953  6955
.LJZ       298   309   311   313   315   415   421   427   433   538   544   550   556  1069
          1157  1244  1353  1526  1691  1858  2021  2188  2271  2363  2456  2551  2648  2899
          2979  2983  3152  3505  3589  3738  3814  [illegible]  4030  4136  4368  4473  4552  4875
          5521  5880  5980  6079  6188  6295  6379  6455  6602  6746  6921
.LLDCT     413   419   425   431   536   542   548   554  1327  1331  1335  1339  1343  1347
          3066  3154  3160  3166  3254  3263  3272  3279  3286  3293  3300  3307  3314  3321
          3328  3335  3342  3349  3356  3363  3370  3377  3384  3391  3480  3591  3598  3605
          3612  3705  3712  3719  3726  3816  4032  4138  4147  4353  4457  4555  4559  4563
          4567  4572  4576  4580  4584  4589  4593  4597  4601  4606  4610  4614  4618  4623
          4627  4631  4635  4640  4644  4648  4652  4657  4661  4665  4669  4674  4678  4682
          4686  4691  4695  4699  4703  4708  4712  4716  4720  4725  4729  4733  4737  4742
          4746  4750  4754  5199  5206  5213  5220  5227  5234  5241  5248  5255  5262  5269
          5276  5283  5290  5297  5304  5311  5318  5325  5332  5339  5346  5353  5360  5367
          5374  5523  5542  5655  5659  5663  5667  5671  5675  5679  5683  5687  5691  5695
          5699  5703  5707  5711  5715  5719  5723  5727  5731  5735  5739  5743  5747  6501
          6503  6505  6507  6648  6650  6652  6654  6748  6923
.LLOOP    4246  4259  4355  4357  4359  4465  6822  6824  6826  6981  6983  6985
.LPUSH    3702  3931  3935  3939  3943  4034  4040  4046  4140  4149  4240  4242  4244  4253
          4255  4257  4343  4345  4347  4349  4351  4449  4451  4453  4455  4459  4853  4855
          4857  4859  4861  4980  4982  4999  5001  5018  5020  5037  5039  5882  5890  5982
          6066  6160  6265  6267  6269  6271  6273  6373  6375  6457  6459  6604  6606  6774
          6949
.LRFCT    4761  4765  4769  4773  4863  4865  4867  4869  5046  5050  5054  5058  5174  5177
          5180  5183  6796  6798  6800  6802
.LRPCT    4461  5525  5529  5533  5537  5544  5546  5548  5550  5636  5640  5644  5648  6485
          6487  6489  6491  6632  6634  6636  6638  6804  6806  6808  6810
.LTWB     5884  5886  5888  5892  5894  5896  5984  6074  6179  6181  6183  6275  6277  6281
          6283  6377  6836  6838  6840  6842
.MCCOF     194   198   309   311   409   415   427   532   538   550   797   803   888   894
           979   985  1159  1246  1356  2752  2764  2860  2985  3156  3162  3168  3256  3265
          3274  3281  3288  3295  3302  3309  3316  3323  3330  3337  3344  3351  3358  3365
          3372  3379  3386  3393  3482  3494  3498  3593  3614  3707  3728  3818  3824  3931
          3939  4034  4036  4040  4042  4046  4048  4140  4142  4149  4151  4242  4244  4246
          4255  4257  4259  4355  4361  4459  4465  4761  4769  4863  4867  5046  5054  5174
          5180  5203  5210  5217  5224  5231  5238  5245  5252  5259  5266  5273  5280  5287
          5294  5301  5308  5315  5322  5329  5336  5343  5350  5357  5364  5371  5378  5525
          5533  5544  5546  5548  5550  5552  5636  5644  5652  5884  5892  5898  5984  6070
          6074  6164  6179  6277  6281  6377  6459  6461  6465  6469  6473  6477  6481  6485
          6489  6493  6497  6501  6505  6509  6511  6517  6519  6521  6606  6608  6612  6616
          6620  6624  6628  6632  6636  6640  6644  6648  6652  6656  6662  6664  6666  6750
          6754  6758  6762  6766  6770  6774  6776  6782  6786  6790  6796  6800  6804  6808
          6812  6814  6818  6822  6828  6832  6836  6840  6845  6925  6929  6933  6937  6941
          6945  6949  6951  6957  6961  6965  6971  6973  6977  6981  6987  6991  6995
.MCCON     196   200   313   315   421   433   544   556   799   801   890   892   981   983
          1358  1360  2756  2760  2875  2890  3486  3490  3600  3607  3714  3721  3820  3822
          3935  3943  4357  4359  4765  4773  4865  4869  5050  5058  5177  5183  5529  5537
          5640  5648  5886  5888  5894  5896  6181  6183  6275  6283  6457  6463  6467  6471
          6475  6479  6483  6487  6491  6495  6499  6503  6507  6513  6515  6524  6604  6610
          6614  6618  6622  6626  6630  6634  6638  6642  6646  6650  6654  6658  6660  6669
          6752  6756  6760  6764  6768  6772  6778  6780  6784  6788  6792  6794  6798  6802
          6806  6810  6816  6820  6824  6826  6830  6834  6838  6842  6927  6931  6935  6939
          6943  6947  6953  6955  6959  6963  6967  6969  6975  6979  6983  6985  6989  6993
.MCENA     801   803   892   894   983   985
.MCJP      797   799   801   803  6469  6471  6473  6475  6616  6618  6620  6622  6766  6768
          6770  6772  6941  6943  6945  6947
.MCJPP    2752  2756  2760  2764  2860  2875  2890  2985  4970  4972  4974  4976  4978  4989
          4991  4993  4995  4997  5008  5010  5012  5014  5016  5027  5029  5031  5033  5035
          6521  6666  6814  6816  6318  6820  6973  6975  6977  6979
.MCJS      979   981   983   985  1067  1155  1159  1240  1329  1333  1337  1341  1345  1349
          1439  1441  1443  1445  1449  1453  1457  1461  1465  1469  1473  1477  1481  1485
          1489  1493  1497  1501  1505  1509  1513  1515  1517  1519  1604  1608  1610  1614
          1618  1622  1626  1630  1634  1638  1642  1646  1650  1654  1658  1662  1666  1670
          1674  1678  1680  1682  1684  1769  1773  1777  1781  1785  1789  1793  1797  1801
          1805  1809  1813  1817  1821  1825  1829  1833  1837  1841  1845  1847  1849  1851   SEQ 0899
          1936  1940  1944  1948  1952  1956  1960  1964  1968  1972  1976  1980  1984  1988
          1992  1996  2000  2004  2008  2012  2014  2016  2018  2099  2103  2107  2111  2115
          2119  2123  2127  2131  2135  2139  2143  2147  2151  2155  2159  2163  2167  2171
          2175  2177  2179  2181  2269  2273  2275  2277  2279  2361  2365  2367  2369  2371
          2454  2458  2460  2462  2464  2549  2553  2555  2557  2559  2646  2650  2652  2654
          2656  2854  2856  2869  2871  2884  2886  2977  2981  3920  3922  3924  3926  4557
          4561  4565  4569  4574  4578  4582  4586  4591  4595  4599  4603  4608  4612  4616
          4620  4625  4629  4633  4637  4642  4646  4650  4654  4659  4663  4667  4671  4676
          4680  4684  4688  4693  4697  4701  4705  4710  4714  4718  4722  4727  4731  4735
          4739  4744  4748  4752  4756  4984  5003  5022  5041  5201  5208  5215  5222  5229
          5236  5243  5250  5257  5264  5271  5278  5285  5292  5299  5306  5313  5320  5327
          5334  5341  5348  5355  5362  5369  5376  6068  6162  6166  6170  6174  6517  6662
          6758  6760  6762  6764  6933  6935  6937  6939
.MCJV      888   890   892   894  6477  6479  6481  6483  6624  6626  6628  6630  6782  6784
          6786  6788  6957  6959  6961  6963
.MCONT     667   669   671   673   677   681   685   689   693   697   701   705   709  1149
          1151  1153  3398  3619  4052  6493  6495  6497  6499  6640  6642  6644  6646  6828
          6830  6832  6834  6987  6989  6991  6993
.MCRTN    1071  1161  1246  1356  1358  1360  1528  1693  1860  2023  2190  2281  2373  2375
          2466  2468  2470  2561  2563  2565  2567  2658  2660  2662  2664  2666  2864  2879
          2894  2987  3709  3716  3723  3730  3733  3933  3937  3941  3945  4370  4470  4871
          4986  5005  5024  5043  6072  6185  6285  6519  6524  6664  6669  6812  6971
.MDISA     194   196   309   313   415   421   538   544   797   799   888   890   979   981
          1356  1358  2752  2756  2860  2875  3482  3486  3593  3600  3707  3714  3818  3820
          3931  3935  4034  4040  4046  4355  4357  4761  4765  4863  4865  5046  5050  5174
          5177  5525  5529  5544  5546  5548  5550  5636  5640  5884  5886  5892  5894  6179
          6181  6281  6283  6461  6463  6469  6471  6477  6479  6485  6487  6493  6495  6501
          6503  6511  6513  6608  6610  6616  6618  6624  6626  6632  6634  6640  6642  6648
          6650  6656  6658  6750  6752  6758  6760  6766  6768  6776  6778  6782  6784  6790
          6792  6796  6798  6804  6806  6814  6816  6822  6824  6828  6830  6836  6838  6925
          6927  6933  6935  6941  6943  6951  6953  6957  6959  6965  6967  6973  6975  6981
          6983  6987  6989
.MENA      198   200   311   315   409   427   433   532   550   556  1159  1246  1360  2760
          2764  2890  2985  3156  3162  3168  3256  3265  3274  3281  3288  3295  3302  3309
          3316  3323  3330  3337  3344  3351  3358  3365  3372  3379  3386  3393  3490  3494
          3498  3607  3614  3721  3728  3822  3824  3939  3943  4036  4042  4048  4140  4142
          4149  4151  4242  4244  4246  4255  4257  4259  4359  4361  4459  4465  4769  4773
          4867  4869  5054  5058  5180  5183  5203  5210  5217  5224  5231  5238  5245  5252
          5259  5266  5273  5280  5287  5294  5301  5308  5315  5322  5329  5336  5343  5350
          5357  5364  5371  5378  5533  5537  5552  5644  5648  5652  5388  5896  5898  5984
          6070  6074  6164  6183  6275  6277  6377  6457  6459  6465  6467  6473  6475  6481
          6483  6489  6491  6497  6499  6505  6507  6509  6515  6517  6519  6521  6524  6604
          6606  6612  6614  6620  6622  6628  6630  6636  6638  6644  6646  6652  6654  6660
          6662  6664  6666  6669  6754  6756  6762  6764  6770  6772  6774  6780  6786  6788
          6794  6800  6802  6808  6810  6812  6818  6820  6826  6832  6834  6840  6842  6845
          6929  6931  6937  6939  6945  6947  6949  6955  6961  6963  6969  6971  6977  6979
          6985  6991  6993  6995
.MJ        194   196   198   200   300   302   304   306   309   311   313   315   409   411
           413   415   417   419   421   423   425   427   429   431   433   436   438   532
           534   536   538   540   542   544   546   548   550   552   554   556   559   561
           665   667   669   671   673   675   677   679   681   683   685   687   689   691
           693   695   697   699   701   703   705   707   709   711   797   799   801   803
```

888
1327
1439
1467
1495
1523
1630
1658
1686
1793
1821
1849
1956
1984
2012
2119
2147
2175
2369
2652
2775
2886
3160
3269
3302
3335
3367
3480
3595
3714
3922
4044
4242
4363
4569
4599
4629
4659
4688
4718
4748
4861
4989
5020
5054
5206
5238
5271
5304
5336
5369
5400
5428

890
1329
1441
1469
1497
1604
1632
1660
1688
1795
1823
1851
1958
1986
2014
2121
2149
2177
2371
2654
2854
2888
3162
3272
3304
3337
3370
3482
3598
3719
3924
4046
4244
4366
4572
4601
4631
4661
4691
4720
4750
4863
4991
5022
5056
5208
5241
5273
5306
5339
5371
5402
5430

892
1331
1443
1471
1499
1606
1634
1662
1769
1797
1825
1853
1960
1988
2016
2123
2151
2179
2454
2656
2856
2890
3164
3274
3307
3339
3372
3484
3600
3721
3926
4048
4246
4457
4574
4603
4633
4663
4693
4722
4752
4865
4993
5027
5058
5210
5243
5276
5308
5341
5374
5404
5432

894
1333
1445
1473
1501
1608
1636
1664
1771
1799
1827
1855
1962
1990
2018
2125
2153
2181
2458
2752
2858
2892
3166
3276
3309
3342
3374
3486
3602
3726
3931
4050
4251
4461
4576
4606
4635
4665
4695
4725
4754
4867
4995
5029
5060
5213
5245
5278
5311
5343
5376
5406
5434

979
1335
1447
1475
1503
1610
1638
1666
1773
1801
1829
1936
1964
1992
2099
2127
2155
2183
2460
2754
2860
2896
3168
3279
3311
3344
3377
3488
3605
3728
3935
4055
4253
4463
4578
4608
4638
4667
4697
4727
4756
4869
4997
5031
5063
5215
5248
5280
5313
5346
5378
5408
5436

981
1337
1449
1477
1505
1612
1640
1668
1775
1803
1831
1938
1966
1994
2101
2129
2157
2185
2462
2756
2862
2977
3170
3281
3314
3346
3379
3490
3607
3735
3939
4138
4255
4465
4580
4610
4640
4669
4699
4729
4759
4873
4999
5033
5174
5217
5250
5283
5315
5348
5381
5410
5523

983
1339
1451
1479
1507
1614
1642
1670
1777
1805
1833
1940
1968
1996
2103
2131
2159
2269
2464
2758
2866
2981
3252
3283
3316
3349
3381
3492
3609
3816
3943
4140
4257
4467
4582
4612
4642
4671
4701
4731
4761
4970
5001
5035
5177
5220
5252
5285
5318
5350
5383
5412
5525

985
1341
1453
1481
1509
1616
1644
1672
1779
1807
1835
1942
1970
1998
2105
2133
2161
2273
2549
2760
2869
2985
3254
3286
3318
3351
3384
3494
3612
3818
3948
4142
4259
4555
4584
4614
4644
4674
4703
4733
4763
4972
5003
5037
5180
5222
5255
5287
5320
5353
5386
5414
5527

1067
1343
1455
1483
1511
1618
1646
1674
1781
1809
1837
1944
1972
2000
2107
2135
2163
2275
2553
2762
2871
2989
3256
3288
3321
3353
3386
3496
3614
3820
4032
4144
4262
4557
4586
4616
4646
4676
4705
4735
4765
4974
5008
5039
5183
5224
5257
5290
5322
5355
5388
5416
5529

1155
1345
1457
1485
1513
1620
1648
1676
1783
1811
1839
1946
1974
2002
2109
2137
2165
2277
2555
2764
2873
3064
3258
3290
3323
3356
3388
3498
3616
3822
4034
4147
4353
4559
4589
4618
4648
4678
4708
4737
4767
4976
5010
5041
5194
5227
5259
5292
5325
5357
5390
5418
5531

1159
1347
1459
1487
1515
1622
1650
1678
1785
1813
1841
1948
1976
2004
2111
2139
2167
2279
2557
2766
2875
3066
3260
3293
3325
3358
3391
3500
3621
3824
4036
4149
4355
4561
4591
4620
4650
4680
4710
4739
4769
4978
5012
5046
5196
5229
5262
5294
5327
5360
5392
5420
5533

1164
1349
1461
1489
1517
1624
1652
1680
1787
1815
1843
1950
1978
2006
2113
2141
2169
2361
2559
2769
2877
3154
3263
3295
3328
3360
3393
3503
3705
3826
4038
4151
4357
4563
4593
4623
4652
4682
4712
4742
4771
4980
5014
5048
5199
5231
5264
5297
5329
5362
5394
5422
5535

1240
1351
1463
1491
1519
1626
1654
1682
1789
1817
1845
1952
1980
2008
2115
2143
2171
2365
2646
2771
2881
3156
3265
3297
3330
3363
3395
3591
3707
3828
4040
4238
4359
4565
4595
4625
4654
4684
4714
4744
4773
4982
5016
5050
5201
5234
5266
5299
5332
5364
5396
5424
5537

1242
1363
1465
1493
1521
1628
1656
1684
1791
1819
1847
1954
1982
2010
2117
2145
2173
2367
2650
2773
2884
3158
3267
3300
3332
3365
3400
3593
3712
3920
4042
4240
4361
4567
4597
4627
4657
4686
4716
4746
4775
4984
5018
5052
5203
5236
5269
5301
5334
5367
5398
5426
5539

```
         5542   5544   5546   5548   5550   5552   5554   5634   5636   5638   5640   5642   5644   5646
         5648   5650   5652   5655   5657   5659   5661   5663   5665   5667   5669   5671   5673   5675
         5677   5679   5681   5683   5685   5687   5689   5691   5693   5695   5697   5699   5701   5703
         5705   5707   5709   5711   5713   5715   5717   5719   5721   5723   5725   5727   5729   5731
         5733   5735   5737   5739   5741   5743   5745   5747   5749   5752   5754   5757   5759   5761
         5763   5765   5767   5769   5771   5773   5775   5777   5779   5781   5783   5785   5787   5789
         5791   5793   5795   5797   5799   5801   5803   5882   5884   5886   5888   5890   5892   5894
         5896   5901   5903   5982   5984   5986   5988   6066   6068   6074   6076   6160   6162   6166
         6168   6170   6172   6174   6176   6179   6181   6183   6265   6267   6269   6271   6273   6275
         6277   6279   6281   6283   6287   6289   6291   6293   6373   6375   6377   6457   6461   6463
         6465   6467   6469   6471   6473   6475   6477   6479   6481   6483   64[illegible]   6487   6489   6491
         6493   6495   6497   6499   6501   6503   6505   6507   6509   6511   651[illegible]   6515   6517   6519
         6521   6527   6604   6608   6610   6612   6614   6616   6618   6620   6622   6624   6626   6628
         6630   6632   6634   6636   6638   6640   6642   6644   6646   6648   6650   6652   6654   6656
         6658   6660   6662   6664   6666   6672   6748   6750   6752   6754   6756   6758   6760   6762
         6764   6766   6768   6770   6772   6774   6776   6778   6780   6782   6784   6786   6788   6790
         6792   6794   6796   6798   6800   6802   6804   6806   6808   6810   6812   6814   6816   6818
         6820   6822   6824   6826   6828   6830   6832   6834   6836   6838   6840   6842   6845   6848
         6923   6925   6927   6929   6931   6933   6935   6937   6939   6941   6943   6945   6947   6949
         6951   6953   6955   6957   6959   6961   6963   6965   6967   6969   6971   6973   6975   6977
         6979   6981   6983   6985   6987   6989   6991   6993   6995   6998   7000   7002
.MJMAP    194    196    198    200    300    302    304    306    411    417    423    429    436    438
          534    540    546    552    559    561    665    675    679    683    687    691    695    699
          703    707    711   1164   1242   1351   1363   1447   1451   1455   1459   1463   1467   1471
         1475   1479   1483   1487   1491   1495   1499   1503   1507   1511   1521   1523   1606   1612
         1616   1620   1624   1628   1632   1636   1640   1644   1648   1652   1656   1660   1664   1668
         1672   1676   1686   1688   1771   1775   1779   1783   1787   1791   1795   1799   1803   1807
         1811   1815   1819   1823   1827   1831   1835   1839   1843   1853   1855   1938   1942   1946
         1950   1954   1958   1962   1966   1970   1974   1978   1982   1986   1990   1994   1998   2002
         2006   2010   2101   2105   2109   2113   2117   2121   2125   2129   2133   2137   2141   2145
         2149   2153   2157   2161   2165   2169   2173   2183   2185   2754   2758   2762   2766   2769
         2771   2773   2775   2858   2862   2866   2873   2877   2881   2888   2892   2896   2989   3064
         3158   3164   3170   3252   3258   3260   3267   3269   3276   3283   3290   3297   3304   3311
         3318   3325   3332   3339   3346   3353   3360   3367   3374   3381   3388   339[illegible]   3400   3484
         3488   3492   3496   3500   3503   3595   3602   3609   3616   3621   3735   3826   3828   3948
         4038   4044   4050   4055   4144   4238   4251   4262   4363   4366   4463   4467   4759   4763
         4767   4771   4775   4873   5048   5052   5056   5060   5063   5194   5196   5381   5383   5386
         5388   5390   5392   5394   5396   5398   5400   5402   5404   5406   5408   5410   5412   5414
         5416   5418   5420   5422   5424   5426   5428   5430   5432   5434   5436   5527   5531   5535
         5539   5554   5634   5638   5642   5646   5650   5657   5661   5665   5669   5673   5677   5681
         5685   5689   5693   5697   5701   5705   5709   5713   5717   5721   5725   5729   5733   5737
         5741   5745   5749   5752   5754   5757   5759   5761   5763   5765   5767   5769   5771   5773
         5775   5777   5779   5781   5783   5785   5787   5789   5791   5793   5795   5797   5799   5801
         5803   5901   5903   5986   5988   6076   6168   6172   6176   6279   6287   6289   6291   6293
         6461   6463   6465   6467   6527   6608   6610   6612   6614   6672   6750   6752   6754   6756
         6848   6925   6927   6929   6931   6998   7000   7002
.MJRP     409    532   3156   3162   3168   3256   3265   3274   3281   3288   3295   3302   3309   3516
         3323   3330   3337   3344   3351   3358   3365   3372   3379   3386   3393   3482   3486   3490
         3494   3498   4036   4042   4048   4142   4151   4361   5203   5210   5217   5224   5231   5238
         5245   5252   5259   5266   5273   5280   5287   5294   5301   5308   5315   5322   5329   5336
         5343   5350   5357   5364   5371   5378   5552   5652   5898   6070   6164   6509   6511   6513
         6515   6656   6658   6660   6790   6792   6794   6845   6965   6967   6969   6995
.MJSRP   3593   3600   3607   3614   3707   3714   3721   3728   3818   3820   3822   3824   6776   6778
```

0889

```
          6780  6951  6953  6955
.MJZ       298   309   311   313   315   415   421   427   433   538   544   550   556  1069    SEQ 0902
          1157  1244  1353  1526  1691  1858  2021  2188  2271  2363  2456  2551  2648  2899
          2979  2983  3152  3505  3589  3738  3814  3928  4030  4136  4368  4473  4552  4875
          5521  5880  5980  6079  6188  6295  6379  6455  6602  6746  6921
.MLDCT     413   419   425   431   536   542   548   554  1327  1331  1335  1339  1343  1347
          3066  3154  3160  3166  3254  3263  3272  3279  3286  3293  3300  3307  3314  3321
          3328  3335  3342  3349  3356  3363  3370  3377  3384  3391  3480  3591  3598  3605
          3612  3705  3712  3719  3726  3816  4032  4138  4147  4353  4457  4555  4559  4563
          4567  4572  4576  4580  4584  4589  4593  4597  4601  4606  4610  4614  4618  4623
          4627  4631  4635  4640  4644  4648  4652  4657  4661  4665  4669  4674  4678  4682
          4686  4691  4695  4699  4703  4708  4712  4716  4720  4725  4729  4733  4737  4742
          4746  4750  4754  5199  5206  5213  5220  5227  5234  5241  5248  5255  5262  5269
          5276  5283  5290  5297  5304  5311  5318  5325  5332  5339  5346  5353  5360  5367
          5374  5523  5542  5655  5659  5663  5667  5671  5675  5679  5683  5687  5691  5695
          5699  5703  5707  5711  5715  5719  5723  5727  5731  5735  5739  5743  5747  6501
          6503  6505  6507  6648  6650  6652  6654  6748  6923
.MLOOP    4246  4259  4355  4357  4359  4465  6822  6824  6826  6981  6983  6985
.MPUSH    3702  3931  3935  3939  3943  4034  4040  4046  4140  4149  4240  4242  4244  4253
          4255  4257  4343  4345  4347  4349  4351  4449  4451  4453  4455  4459  4853  4855
          4857  4859  4861  4980  4982  4999  5001  5018  5020  5037  5039  5882  5890  5982
          6066  6160  6265  6267  6269  6271  6273  6373  6375  6457  6459  6604  6606  6774
          6949
.MRFCT    4761  4765  4769  4773  4863  4865  4867  4869  5046  5050  5054  5058  5174  5177
          5180  5183  6796  6798  6800  6802
.MRPCT    4461  5525  5529  5533  5537  5544  5546  5548  5550  5636  5640  5644  5648  6485
          6487  6489  6491  6632  6634  6636  6638  6804  6806  6808  6810
.MTWB     5884  5886  5888  5892  5894  5896  5984  6074  6179  6181  6183  6275  6277  6281
          6283  6377  6836  6838  6840  6842
.RCCOF     194   198   309   311   409   415   427   532   538   550   797   803   888   894
           979   985  1159  1246  1356  2752  2764  2860  2985  3156  3162  3168  3256  3265
          3274  3281  3288  3295  3302  3309  3316  3323  3330  3337  3344  3351  3358  3365
          3372  3379  3386  3393  3482  3494  3498  3593  3614  3707  3728  3818  3824  3931
          3939  4034  4036  4040  4042  4046  4048  4140  4142  4149  4151  4242  4244  4246
          4255  4257  4259  4355  4361  4459  4465  4761  4769  4863  4867  5046  5054  5174
          5180  5203  5210  5217  5224  5231  5238  5245  5252  5259  5266  5273  5280  5287
          5294  5301  5308  5315  5322  5329  5336  5343  5350  5357  5364  5371  5378  5525
          5533  5544  5546  5548  5550  5552  5636  5644  5652  5884  5892  5898  5984  6070
          6074  6164  6179  6277  6281  6377  6459  6461  6465  6469  6473  6477  6481  6485
          6489  6493  6497  6501  6505  6509  6511  6517  6519  6521  6606  6608  6612  6616
          6620  6624  6628  6632  6636  6640  6644  6648  6652  6656  6662  6664  6666  6750
          6754  6758  6762  6766  6770  6774  6776  6782  6786  6790  6796  6800  6804  6808
          6812  6814  6818  6822  6828  6832  6836  6840  6845  6925  6929  6933  6937  6941
          6945  6949  6951  6957  6961  6965  6971  6973  6977  6981  6987  6991  6995
.RCCON     196   200   313   315   421   433   544   556   799   801   890   892   981   983
          1358  1360  2756  2760  2875  2890  3486  3490  3600  3607  3714  3721  3820  3822
          3935  3943  4357  4359  4765  4773  4865  4869  5050  5058  5177  5183  5529  5537
          5640  5648  5886  5888  5894  5896  6181  6183  6275  6283  6457  6463  6467  6471
          6475  6479  6483  6487  6491  6495  6499  6503  6507  6513  6515  6524  6604  6610
          6614  6618  6622  6626  6630  6634  6638  6642  6646  6650  6654  6658  6660  6669
          6752  6756  6760  6764  6768  6772  6778  6780  6784  6788  6792  6794  6798  6802
          6806  6810  6816  6820  6824  6826  6830  6834  6838  6842  6927  6931  6935  6939
          6943  6947  6953  6955  6959  6963  6967  6969  6975  6979  6983  6985  6989  6993
```

MWO

```
.RCENA    801   803   892   894   983   985
.RCJP     797   799   801   803  6469  6471  6473  6475  6616  6618  6620  6622  6766  6768
         6770  6772  6941  6943  6945  6947
.RCJPP   2752  2756  2760  2764  2860  2875  2890  2985  4970  4972  4974  4976  4978  4989
         4991  4993  4995  4997  5008  5010  5012  5014  5016  5027  5029  5031  5033  5035
         6521  6666  6814  6816  6818  6820  6973  6975  6977  6979
.RCJS     979   981   983   985  1067  1155  1159  1240  1329  1333  1337  1341  1345  1349
         1439  1441  1443  1445  1449  1453  1457  1461  1465  1469  1473  1477  1481  1485
         1489  1493  1497  1501  1505  1509  1513  1515  1517  1519  1604  1608  1610  1614
         1618  1622  1626  1630  1634  1638  1642  1646  1650  1654  1658  1662  1666  1670
         1674  1678  1680  1682  1684  1769  1773  1777  1781  1785  1789  1793  1797  1801
         1805  1809  1813  1817  1821  1825  1829  1833  1837  1841  1845  1847  1849  1851
         1936  1940  1944  1948  1952  1956  1960  1964  1968  1972  1976  1980  1984  1988
         1992  1996  2000  2004  2008  2012  2014  2016  2018  2099  2103  2107  2111  2115
         2119  2123  2127  2131  2135  2139  2143  2147  2151  2155  2159  2163  2167  2171
         2175  2177  2179  2181  2269  2273  2275  2277  2279  2361  2365  2367  2369  2371
         2454  2458  2460  2462  2464  2549  2553  2555  2557  2559  2646  2650  2652  2654
         2656  2854  2856  2869  2871  2884  2886  2977  2981  3920  3922  3924  3926  4557
         4561  4565  4569  4574  4578  4582  4586  4591  4595  4599  4603  4608  4612  4616
         4620  4625  4629  4633  4637  4642  4646  4650  4654  4659  4663  4667  4671  4676
         4680  4684  4688  4693  4697  4701  4705  4710  4714  4718  4722  4727  4731  4735
         4739  4744  4748  4752  4756  4984  5003  5022  5041  5201  5208  5215  5222  5229
         5236  5243  5250  5257  5264  5271  5278  5285  5292  5299  5306  5313  5320  5327
         5334  5341  5348  5355  5362  5369  5376  6068  6162  6166  6170  6174  6517  6662
         6758  6760  6762  6764  6933  6935  6937  6939
.RCJV     888   890   892   894  6477  6479  6481  6483  6624  6626  6628  6630  6782  6784
         6786  6788  6957  6959  6961  6963
.RCONT    667   669   671   673   677   681   685   689   693   697   701   705   709  1149
         1151  1153  3398  3619  4052  6493  6495  6497  6499  6640  6642  6644  6646  6828
         6830  6832  6834  6987  6989  6991  6993
.RCRTN   1071  1161  1246  1356  1358  1360  1528  1693  1860  2023  2190  2281  2373  2375
         2466  2468  2470  2561  2563  2565  2567  2658  2660  2662  2664  2666  2864  2879
         2894  2987  3709  3716  3723  3730  3733  3933  3937  3941  3945  4370  4470  4871
         4986  5005  5024  5043  6072  6185  6285  6519  6524  6664  6669  6812  6971
.RDISA    194   196   309   313   415   421   538   544   797   799   888   890   979   981
         1356  1358  2752  2756  2860  2875  3482  3486  3593  3600  3707  3714  3818  3820
         3931  3935  4034  4040  4046  4355  4357  4761  4765  4863  4865  5046  5050  5174
         5177  5525  5529  5544  5546  5548  5550  5636  5640  5884  5886  5892  5894  6179
         6181  6281  6283  6461  6463  6469  6471  6477  6479  6485  6487  6493  6495  6501
         6503  6511  6513  6608  6610  6616  6618  6624  6626  6632  6634  6640  6642  6648
         6650  6656  6658  6750  6752  6758  6760  6766  6768  6776  6778  6782  6784  6790
         6792  6796  6798  6804  6806  6814  6816  6822  6824  6828  6830  6836  6838  6925
         6927  6933  6935  6941  6943  6951  6953  6957  6959  6965  6967  6973  6975  6981
         6983  6987  6989
.RENA     198   200   311   315   409   427   433   532   550   556  1159  1246  1360  2760
         2764  2890  2985  3156  3162  3168  3256  3265  3274  3281  3288  3295  3302  3309
         3316  3323  3330  3337  3344  3351  3358  3365  3372  3379  3386  3393  3490  3494
         3498  3607  3614  3721  3728  3822  3824  3939  3943  4036  4042  4048  4140  4142
         4149  4151  4242  4244  4246  4255  4257  4259  4359  4361  4459  4465  4769  4773
         4867  4869  5054  5058  5180  5183  5203  5210  5217  5224  5231  5238  5245  5252
         5259  5266  5273  5280  5287  5294  5301  5308  5315  5322  5329  5336  5343  5350
         5357  5364  5371  5378  5533  5537  5552  5644  5648  5652  5888  5896  5898  5984
         6070  6074  6164  6183  6275  6277  6377  6457  6459  6465  6467  6473  6475  6481
```

```
    6483   6489   6491   6497   6499   6505   6507   6509   6515   6517   6519   6521   6524   6604   SEQ 0904
    6606   6612   6614   6620   6622   6628   6630   6636   6638   6644   6646   6652   6654   6660
    6662   6664   6666   6669   6754   6756   6762   6764   6770   6772   6774   6780   6786   6788
    6794   6800   6802   6808   6810   6812   6818   6820   6826   6832   6834   6840   6842   6845
    6929   6931   6937   6939   6945   6947   6949   6955   6961   6963   6969   6971   6977   6979
    6985   6991   6993   6995
.RJ  194    196    198    200    300    302    304    306    309    311    313    315    409    411
     413    415    417    419    421    423    425    427    429    431    433    436    438    532
     534    536    538    540    542    544    546    548    550    552    554    556    559    561
     665    667    669    671    673    675    677    679    681    683    685    687    689    691
     693    695    697    699    701    703    705    707    709    711    797    799    801    803
     888    890    892    894    979    981    983    985   1067   1155   1159   1164   1240   1242
    1327   1329   1331   1333   1335   1337   1339   1341   1343   1345   1347   1349   1351   1363
    1439   1441   1443   1445   1447   1449   1451   1453   1455   1457   1459   1461   1463   1465
    1467   1469   1471   1473   1475   1477   1479   1481   1483   1485   1487   1489   1491   1493
    1495   1497   1499   1501   1503   1505   1507   1509   1511   1513   1515   1517   1519   1521
    1523   1604   1606   1608   1610   1612   1614   1616   1618   1620   1622   1624   1626   1628
    1630   1632   1634   1636   1638   1640   1642   1644   1646   1648   1650   1652   1654   1656
    1658   1660   1662   1664   1666   1668   1670   1672   1674   1676   1678   1680   1682   1684
    1686   1688   1769   1771   1773   1775   1777   1779   1781   1783   1785   1787   1789   1791
    1793   1795   1797   1799   1801   1803   1805   1807   1809   1811   1813   1815   1817   1819
    1821   1823   1825   1827   1829   1831   1833   1835   1837   1839   1841   1843   1845   1847
    1849   1851   1853   1855   1936   1938   1940   1942   1944   1946   1948   1950   1952   1954
    1956   1958   1960   1962   1964   1966   1968   1970   1972   1974   1976   1978   1980   1982
    1984   1986   1988   1990   1992   1994   1996   1998   2000   2002   2004   2006   2008   2010
    2012   2014   2016   2018   2099   2101   2103   2105   2107   2109   2111   2113   2115   2117
    2119   2121   2123   2125   2127   2129   2131   2133   2135   2137   2139   2141   2143   2145
    2147   2149   2151   2153   2155   2157   2159   2161   2163   2165   2167   2169   2171   2173
    2175   2177   2179   2181   2183   2185   2269   2273   2275   2277   2279   2361   2365   2367
    2369   2371   2454   2458   2460   2462   2464   2549   2553   2555   2557   2559   2646   2650
    2652   2654   2656   2752   2754   2756   2758   2760   2762   2764   2766   2769   2771   2773
    2775   2854   2856   2858   2860   2862   2866   2869   2871   2873   2875   2877   2881   2884
    2886   2888   2890   2892   2896   2977   2981   2985   2989   3064   3066   3154   3156   3158
    3160   3162   3164   3166   3168   3170   3252   3254   3256   3258   3260   3263   3265   3267
    3269   3272   3274   3276   3279   3281   3283   3286   3288   3290   3293   3295   3297   3300
    3302   3304   3307   3309   3311   3314   3316   3318   3321   3323   3325   3328   3330   3332
    3335   3337   3339   3342   3344   3346   3349   3351   3353   3356   3358   3360   3363   3365
    3367   3370   3372   3374   3377   3379   3381   3384   3386   3388   3391   3393   3395   3400
    3480   3482   3484   3486   3488   3490   3492   3494   3496   3498   3500   3503   3591   3593
    3595   3598   3600   3602   3605   3607   3609   3612   3614   3616   3621   3705   3707   3712
    3714   3719   3721   3726   3728   3735   3816   3818   3820   3822   3824   3826   3828   3920
    3922   3924   3926   3931   3935   3939   3943   3948   4032   4034   4036   4038   4040   4042
    4044   4046   4048   4050   4055   4138   4140   4142   4144   4147   4149   4151   4238   4240
    4242   4244   4246   4251   4253   4255   4257   4259   4262   4353   4355   4357   4359   4361
    4363   4366   4457   4461   4463   4465   4467   4555   4557   4559   4561   4563   4565   4567
    4569   4572   4574   4576   4578   4580   4582   4584   4586   4589   4591   4593   4595   4597
    4599   4601   4603   4606   4608   4610   4612   4614   4616   4618   4620   4623   4625   4627
    4629   4631   4633   4635   4637   4640   4642   4644   4646   4648   4650   4652   4654   4657
    4659   4661   4663   4665   4667   4669   4671   4674   4676   4678   4680   4682   4684   4686
    4688   4691   4693   4695   4697   4699   4701   4703   4705   4708   4710   4712   4714   4716
    4718   4720   4722   4725   4727   4729   4731   4733   4735   4737   4739   4742   4744   4746
    4748   4750   4752   4754   4756   4759   4761   4763   4765   4767   4769   4771   4773   4775
    4861   4863   4865   4867   4869   4873   4970   4972   4974   4976   4978   4980   4982   4984
```

```
             4989   4991   4993   4995   4997   4999   5001   5003   5008   5010   5012   5014   5016   5018      SEQ 0905
             5020   5022   5027   5029   5031   5033   5035   5037   5039   5041   5046   5048   5050   5052
             5054   5056   5058   5060   5063   5174   5177   5180   5183   5194   5196   5199   5201   5203
             5206   5208   5210   5213   5215   5217   5220   5222   5224   5227   5229   5231   5234   5236
             5238   5241   5243   5245   5248   5250   5252   5255   5257   5259   5262   5264   5266   5269
             5271   5273   5276   5278   5280   5283   5285   5287   5290   5292   5294   5297   5299   5301
             5304   5306   5308   5311   5313   5315   5318   5320   5322   5325   5327   5329   5332   5334
             5336   5339   5341   5343   5346   5348   5350   5353   5355   5357   5360   5362   5364   5367
             5369   5371   5374   5376   5378   5381   5383   5386   5388   5390   5392   5394   5396   5398
             5400   5402   5404   5406   5408   5410   5412   5414   5416   5418   5420   5422   5424   5426
             5428   5430   5432   5434   5436   5523   5525   5527   5529   5531   5533   5535   5537   5539
             5542   5544   5546   5548   5550   5552   5554   5634   5636   5638   5640   5642   5644   5646
             5648   5650   5652   5655   5657   5659   5661   5663   5665   5667   5669   5671   5673   5675
             5677   5679   5681   5683   5685   5687   5689   5691   5693   5695   5697   5699   5701   5703
             5705   5707   5709   5711   5713   5715   5717   5719   5721   5723   5725   5727   5729   5731
             5733   5735   5737   5739   5741   5743   5745   5747   5749   5752   5754   5757   5759   5761
             5763   5765   5767   5769   5771   5773   5775   5777   5779   5781   5783   5785   5787   5789
             5791   5793   5795   5797   5799   5801   5803   5882   5884   5886   5888   5890   5892   5894
             5896   5901   5903   5982   5984   5986   5988   6066   6068   6074   6076   6160   6162   6166
             6168   6170   6172   6174   6176   6179   6181   6183   6265   6267   6269   6271   6273   6275
             6277   6279   6281   6283   6287   6289   6291   6293   6373   6375   6377   6457   6461   6463
             6465   6467   6469   6471   6473   6475   6477   6479   6481   6483   6485   6487   6489   6491
             6493   6495   6497   6499   6501   6503   6505   6507   6509   6511   6513   6515   6517   6519
             6521   6527   6604   6608   6610   6612   6614   6616   6618   6620   6622   6624   6626   6628
             6630   6632   6634   6636   6638   6640   6642   6644   6646   6648   6650   6652   6654   6656
             6658   6660   6662   6664   6666   6672   6748   6750   6752   6754   6756   6758   6760   6762
             6764   6766   6768   6770   6772   6774   6776   6778   6780   6782   6784   6786   6788   6790
             6792   6794   6796   6798   6800   6802   6804   6806   6808   6810   6812   6814   6816   6818
             6820   6822   6824   6826   6828   6830   6832   6834   6836   6838   6840   6842   6845   6848
             6923   6925   6927   6929   6931   6933   6935   6937   6939   6941   6943   6945   6947   6949
             6951   6953   6955   6957   6959   6961   6963   6965   6967   6969   6971   6973   6975   6977
             6979   6981   6983   6985   6987   6989   6991   6993   6995   6998   7000   7002
.RJMAP        194    196    198    200    300    302    304    306    411    417    423    429    436    438
              534    540    546    552    559    561    665    675    679    683    687    691    695    699
              703    707    711   1164   1242   1351   1363   1447   1451   1455   1459   1463   1467   1471
             1475   1479   1483   1487   1491   1495   1499   1503   1507   1511   1521   1523   1606   1612
             1616   1620   1624   1628   1632   1636   1640   1644   1648   1652   1656   1660   1664   1668
             1672   1676   1686   1688   1771   1775   1779   1783   1787   1791   1795   1799   1803   1807
             1811   1815   1819   1823   1827   1831   1835   1839   1843   1853   1855   1938   1942   1946
             1950   1954   1958   1962   1966   1970   1974   1978   1982   1986   1990   1994   1998   2002
             2006   2010   2101   2105   2109   2113   2117   2121   2125   2129   2133   2137   2141   2145
             2149   2153   2157   2161   2165   2169   2173   2183   2185   2754   2758   2762   2766   2769
             2771   2773   2775   2858   2862   2866   2873   2877   2881   2888   2892   2896   2989   3064
             3158   3164   3170   3252   3258   3260   3267   3269   3276   3283   3290   3297   3304   3311
             3318   3325   3332   3339   3346   3353   3360   3367   3374   3381   3388   3395   3400   3484
             3488   3492   3496   3500   3503   3595   3602   3609   3616   3621   3735   3826   3828   3948
             4038   4044   4050   4055   4144   4238   4251   4262   4363   4366   4463   4467   4759   4763
             4767   4771   4775   4873   5048   5052   5056   5060   5063   5194   5196   5381   5383   5386
             5388   5390   5392   5394   5396   5398   5400   5402   5404   5406   5408   5410   5412   5414
             5416   5418   5420   5422   5424   5426   5428   5430   5432   5434   5436   5527   5531   5535
             5539   5554   5634   5638   5642   5646   5650   5657   5661   5665   5669   5673   5677   5681
             5685   5689   5693   5697   5701   5705   5709   5713   5717   5721   5725   5729   5733   5737
             5741   5745   5749   5752   5754   5757   5759   5761   5763   5765   5767   5769   5771   5773
```

```
         5775  5777  5779  5781  5783  5785  5787  5789  5791  5793  5795  5797  5799  5801
         5803  5901  5903  5986  5988  6076  6168  6172  6176  6279  6287  6289  6291  6293
         6461  6463  6465  6467  6527  6608  6610  6612  6614  6672  6750  6752  6754  6756
         6848  6925  6927  6929  6931  6998  7000  7002
.RJRP     409   532  3156  3162  3168  3256  3265  3274  3281  3288  3295  3302  3309  3316
         3323  3330  3337  3344  3351  3358  3365  3372  3379  3386  3393  3482  3486  3490
         3494  3498  4036  4042  4048  4142  4151  4361  5203  5210  5217  5224  5231  5238
         5245  5252  5259  5266  5273  5280  5287  5294  5301  5308  5315  5322  5329  5336
         5343  5350  5357  5364  5371  5378  5552  5652  5898  6070  6164  6509  6511  6513
         6515  6656  6658  6660  6790  6792  6794  6845  6965  6967  6969  6995
.RJSRP   3593  3600  3607  3614  3707  3714  3721  3728  3818  3820  3822  3824  6776  6778
         6780  6951  6953  6955
.RJZ      298   309   311   313   315   415   421   427   433   538   544   550   556  1069
         1157  1244  1353  1526  1691  1858  2021  2188  2271  2363  2456  2551  2648  2899
         2979  2983  3152  3505  3589  3738  3814  3928  4030  4136  4368  4473  4552  4875
         5521  5880  5980  6079  6188  6295  6379  6455  6602  6746  6921
.RLDCT    413   419   425   431   536   542   548   554  1327  1331  1335  1339  1343  1347
         3066  3154  3160  3166  3254  3263  3272  3279  3286  3293  3300  3307  3314  3321
         3328  3335  3342  3349  3356  3363  3370  3377  3384  3391  3480  3591  3598  3605
         3612  3705  3712  3719  3726  3816  4032  4138  4147  4353  4457  4555  4559  4563
         4567  4572  4576  4580  4584  4589  4593  4597  4601  4606  4610  4614  4618  4623
         4627  4631  4635  4640  4644  4648  4652  4657  4661  4665  4669  4674  4678  4682
         4686  4691  4695  4699  4703  4708  4712  4716  4720  4725  4729  4733  4737  4742
         4746  4750  4754  5199  5206  5213  5220  5227  5234  5241  5248  5255  5262  5269
         5276  5283  5290  5297  5304  5311  5318  5325  5332  5339  5346  5353  5360  5367
         5374  5523  5542  5655  5659  5663  5667  5671  5675  5679  5683  5687  5691  5695
         5699  5703  5707  5711  5715  5719  5723  5727  5731  5735  5739  5743  5747  6501
         6503  6505  6507  6648  6650  6652  6654  6748  6923
.RLOOP   4246  4259  4355  4357  4359  4465  6822  6824  6826  6981  6983  6985
.RPUSH   3702  3931  3935  3939  3943  4034  4040  4046  4140  4149  4240  4242  4244  4253
         4255  4257  4343  4345  4347  4349  4351  4449  4451  4453  4455  4459  4853  4855
         4857  4859  4861  4980  4982  4999  5001  5018  5020  5037  5039  5882  5890  5982
         6066  6160  6265  6267  6269  6271  6273  6373  6375  6457  6459  6604  6606  6774
         6949
.RRFCT   4761  4765  4769  4773  4863  4865  4867  4869  5046  5050  5054  5058  5174  5177
         5180  5183  6796  6798  6800  6802
.RRPCT   4461  5525  5529  5533  5537  5544  5546  5548  5550  5636  5640  5644  5648  6485
         6487  6489  6491  6632  6634  6636  6638  6804  6806  6808  6810
.RTWB    5884  5886  5888  5892  5894  5896  5984  6074  6179  6181  6183  6275  6277  6281
         6283  6377  6836  6838  6840  6842
```

```
CALC    194   196   198   200   298   300   302   304   306   309   311   313   315   409    SEQ 0907
        411   413   415   417   419   421   423   425   427   429   431   433   436   438
        532   534   536   538   540   542   544   546   548   550   552   554   556   559
        561   665   667   669   671   673   675   677   679   681   683   685   687   689
        691   693   695   697   699   701   703   705   707   709   711   797   799   801
        803   888   890   892   894   979   981   983   985  1067  1069  1071  1149  1151
       1153  1155  1157  1159  1161  1164  1240  1242  1244  1246  1327  1329  1331  1333
       1335  1337  1339  1341  1343  1345  1347  1349  1351  1353  1356  1358  1360  1363
       1439  1441  1443  1445  1447  1449  1451  1453  1455  1457  1459  1461  1463  1465
       1467  1469  1471  1473  1475  1477  1479  1481  1483  1485  1487  1489  1491  1493
       1495  1497  1499  1501  1503  1505  1507  1509  1511  1513  1515  1517  1519  1521
       1523  1526  1528  1604  1606  1608  1610  1612  1614  1616  1618  1620  1622  1624
       1626  1628  1630  1632  1634  1636  1638  1640  1642  1644  1646  1648  1650  1652
       1654  1656  1658  1660  1662  1664  1666  1668  1670  1672  1674  1676  1678  1680
       1682  1684  1686  1688  1691  1693  1769  1771  1773  1775  1777  1779  1781  1783
       1785  1787  1789  1791  1793  1795  1797  1799  1801  1803  1805  1807  1809  1811
       1813  1815  1817  1819  1821  1823  1825  1827  1829  1831  1833  1835  1837  1839
       1841  1843  1845  1847  1849  1851  1853  1855  1858  1860  1936  1938  1940  1942
       1944  1946  1948  1950  1952  1954  1956  1958  1960  1962  1964  1966  1968  1970
       1972  1974  1976  1978  1980  1982  1984  1986  1988  1990  1992  1994  1996  1998
       2000  2002  2004  2006  2008  2010  2012  2014  2016  2018  2021  2023  2099  2101
       2103  2105  2107  2109  2111  2113  2115  2117  2119  2121  2123  2125  2127  2129
       2131  2133  2135  2137  2139  2141  2143  2145  2147  2149  2151  2153  2155  2157
       2159  2161  2163  2165  2167  2169  2171  2173  2175  2177  2179  2181  2183  2185
       2188  2190  2269  2271  2273  2275  2277  2279  2281  2361  2363  2365  2367  2369
       2371  2373  2375  2454  2456  2458  2460  2462  2464  2466  2468  2470  2549  2551
       2553  2555  2557  2559  2561  2563  2565  2567  2646  2648  2650  2652  2654  2656
       2658  2660  2662  2664  2666  2752  2754  2756  2758  2760  2762  2764  2766  2769
       2771  2773  2775  2854  2856  2858  2860  2862  2864  2866  2869  2871  2873  2875
       2877  2879  2881  2884  2886  2888  2890  2892  2894  2896  2899  2977  2979  2981
       2983  2985  2987  2989  3064  3066  3152  3154  3156  3158  3160  3162  3164  3166
       3168  3170  3252  3254  3256  3258  3260  3263  3265  3267  3269  3272  3274  3276
       3279  3281  3283  3286  3288  3290  3293  3295  3297  3300  3302  3304  3307  3309
       3311  3314  3316  3318  3321  3323  3325  3328  3330  3332  3335  3337  3339  3342
       3344  3346  3349  3351  3353  3356  3358  3360  3363  3365  3367  3370  3372  3374
       3377  3379  3381  3384  3386  3388  3391  3393  3395  3398  3400  3480  3482  3484
       3486  3488  3490  3492  3494  3496  3498  3500  3503  3505  3589  3591  3593  3595
       3598  3600  3602  3605  3607  3609  3612  3614  3616  3619  3621  3702  3705  3707
       3709  3712  3714  3716  3719  3721  3723  3726  3728  3730  3733  3735  3738  3814
       3816  3818  3820  3822  3824  3826  3828  3920  3922  3924  3926  3928  3931  3933
       3935  3937  3939  3941  3943  3945  3948  4030  4032  4034  4036  4038  4040  4042
       4044  4046  4048  4050  4052  4055  4136  4138  4140  4142  4144  4147  4149  4151
       4238  4240  4242  4244  4246  4251  4253  4255  4257  4259  4262  4343  4345  4347
       4349  4351  4353  4355  4357  4359  4361  4363  4366  4368  4370  4449  4451  4453
       4455  4457  4459  4461  4463  4465  4467  4470  4473  4552  4555  4557  4559  4561
       4563  4565  4567  4569  4572  4574  4576  4578  4580  4582  4584  4586  4589  4591
       4593  4595  4597  4599  4601  4603  4606  4608  4610  4612  4614  4616  4618  4620
       4623  4625  4627  4629  4631  4633  4635  4637  4640  4642  4644  4646  4648  4650
       4652  4654  4657  4659  4661  4663  4665  4667  4669  4671  4674  4676  4678  4680
       4682  4684  4686  4688  4691  4693  4695  4697  4699  4701  4703  4705  4708  4710
       4712  4714  4716  4718  4720  4722  4725  4727  4729  4731  4733  4735  4737  4739
       4742  4744  4746  4748  4750  4752  4754  4756  4759  4761  4763  4765  4767  4769
       4771  4773  4775  4853  4855  4857  4859  4861  4863  4865  4867  4869  4871  4873
```

```
SEQ 0895    4875  4970  4972  4974  4976  4978  4980  4982  4984  4986  4989  4991  4993  4995    SEQ 0908
            4997  4999  5001  5003  5005  5008  5010  5012  5014  5016  5018  5020  5022  5024
            5027  5029  5031  5033  5035  5037  5039  5041  5043  5046  5048  5050  5052  5054
            5056  5058  5060  5063  5174  5177  5180  5183  5194  5196  5199  5201  5203  5206
            5208  5210  5213  5215  5217  5220  5222  5224  5227  5229  5231  5234  5236  5238
            5241  5243  5245  5248  5250  5252  5255  5257  5259  5262  5264  5266  5269  5271
            5273  5276  5278  5280  5283  5285  5287  5290  5292  5294  5297  5299  5301  5304
            5306  5308  5311  5313  5315  5318  5320  5322  5325  5327  5329  5332  5334  5336
            5339  5341  5343  5346  5348  5350  5353  5355  5357  5360  5362  5364  5367  5369
            5371  5374  5376  5378  5381  5383  5386  5388  5390  5392  5394  5396  5398  5400
            5402  5404  5406  5408  5410  5412  5414  5416  5418  5420  5422  5424  5426  5428
            5430  5432  5434  5436  5521  5523  5525  5527  5529  5531  5533  5535  5537  5539
            5542  5544  5546  5548  5550  5552  5554  5634  5636  5638  5640  5642  5644  5646
            5648  5650  5652  5655  5657  5659  5661  5663  5665  5667  5669  5671  5673  5675
            5677  5679  5681  5683  5685  5687  5689  5691  5693  5695  5697  5699  5701  5703
            5705  5707  5709  5711  5713  5715  5717  5719  5721  5723  5725  5727  5729  5731
            5733  5735  5737  5739  5741  5743  5745  5747  5749  5752  5754  5757  5759  5761
            5763  5765  5767  5769  5771  5773  5775  5777  5779  5781  5783  5785  5787  5789
            5791  5793  5795  5797  5799  5801  5803  5880  5882  5884  5886  5888  5890  5892
            5894  5896  5898  5901  5903  5980  5982  5984  5986  5988  6066  6068  6070  6072
            6074  6076  6079  6160  6162  6164  6166  6168  6170  6172  6174  6176  6179  6181
            6183  6185  6188  6265  6267  6269  6271  6273  6275  6277  6279  6281  6283  6285
            6287  6289  6291  6293  6295  6373  6375  6377  6379  6455  6457  6459  6461  6463
            6465  6467  6469  6471  6473  6475  6477  6479  6481  6483  6485  6487  6489  6491
            6493  6495  6497  6499  6501  6503  6505  6507  6509  6511  6513  6515  6517  6519
            6521  6524  6527  6602  6604  6606  6608  6610  6612  6614  6616  6618  6620  6622
            6624  6626  6628  6630  6632  6634  6636  6638  6640  6642  6644  6646  6648  6650
            6652  6654  6656  6658  6660  6662  6664  6666  6669  6672  6746  6748  6750  6752
            6754  6756  6758  6760  6762  6764  6766  6768  6770  6772  6774  6776  6778  6780
            6782  6784  6786  6788  6790  6792  6794  6796  6798  6800  6802  6804  6806  6808
            6810  6812  6814  6816  6818  6820  6822  6824  6826  6828  6830  6832  6834  6836
            6838  6840  6842  6845  6848  6921  6923  6925  6927  6929  6931  6933  6935  6937
            6939  6941  6943  6945  6947  6949  6951  6953  6955  6957  6959  6961  6963  6965
            6967  6969  6971  6973  6975  6977  6979  6981  6983  6985  6987  6989  6991  6993
            6995  6998  7000  7002
CONCAT       194   196   198   200   298   300   302   304   306   309   311   313   315   409
             411   413   415   417   419   421   423   425   427   429   431   433   436   438
             532   534   536   538   540   542   544   546   548   550   552   554   556   559
             561   665   667   669   671   673   675   677   679   681   683   685   687   689
             691   693   695   697   699   701   703   705   707   709   711   797   799   801
             803   888   890   892   894   979   981   983   985  1067  1069  1071  1149  1151
            1153  1155  1157  1159  1161  1164  1240  1242  1244  1246  1327  1329  1331  1333
            1335  1337  1339  1341  1343  1345  1347  1349  1351  1353  1356  1358  1360  1363
            1439  1441  1443  1445  1447  1449  1451  1453  1455  1457  1459  1461  1463  1465
            1467  1469  1471  1473  1475  1477  1479  1481  1483  1485  1487  1489  1491  1493
            1495  1497  1499  1501  1503  1505  1507  1509  1511  1513  1515  1517  1519  1521
            1523  1526  1528  1604  1606  1608  1610  1612  1614  1616  1618  1620  1622  1624
            1626  1628  1630  1632  1634  1636  1638  1640  1642  1644  1646  1648  1650  1652
            1654  1656  1658  1660  1662  1664  1666  1668  1670  1672  1674  1676  1678  1680
            1682  1684  1686  1688  1691  1693  1769  1771  1773  1775  1777  1779  1781  1783
            1785  1787  1789  1791  1793  1795  1797  1799  1801  1803  1805  1807  1809  1811
            1813  1815  1817  1819  1821  1823  1825  1827  1829  1831  1833  1835  1837  1839
            1841  1843  1845  1847  1849  1851  1853  1855  1858  1860  1936  1938  1940  1942
```

SEQ 0896 SEQ 0909

1944	1946	1948	1950	1952	1954	1956	1958	1960	1962	1964	1966	1968	1970
1972	1974	1976	1978	1980	1982	1984	1986	1988	1990	1992	1994	1996	1998
2000	2002	2004	2006	2008	2010	2012	2014	2016	2018	2021	2023	2099	2101
2103	2105	2107	2109	2111	2113	2115	2117	2119	2121	2123	2125	2127	2129
2131	2133	2135	2137	2139	2141	2143	2145	2147	2149	2151	2153	2155	2157
2159	2161	2163	2165	2167	2169	2171	2173	2175	2177	2179	2181	2183	2185
2188	2190	2269	2271	2273	2275	2277	2279	2281	2361	2363	2365	2367	2369
2371	2373	2375	2454	2456	2458	2460	2462	2464	2466	2468	2470	2549	2551
2553	2555	2557	2559	2561	2563	2565	2567	2646	2648	2650	2652	2654	2656
2658	2660	2662	2664	2666	2752	2754	2756	2758	2760	2762	2764	2766	2769
2771	2773	2775	2854	2856	2858	2860	2862	2864	2866	2869	2871	2873	2875
2877	2879	2881	2884	2886	2888	2890	2892	2894	2896	2899	2977	2979	2981
2983	2985	2987	2989	3064	3066	3152	3154	3156	3158	3160	3162	3164	3166
3168	3170	3252	3254	3256	3258	3260	3263	3265	3267	3269	3272	3274	3276
3279	3281	3283	3286	3288	3290	3293	3295	3297	3300	3302	3304	3307	3309
3311	3314	3316	3318	3321	3323	3325	3328	3330	3332	3335	3337	3339	3342
3344	3346	3349	3351	3353	3356	3358	3360	3363	3365	3367	3370	3372	3374
3377	3379	3381	3384	3386	3388	3391	3393	3395	3398	3400	3480	3482	3484
3486	3488	3490	3492	3494	3496	3498	3500	3503	3505	3589	3591	3593	3595
3598	3600	3602	3605	3607	3609	3612	3614	3616	3619	3621	3702	3705	3707
3709	3712	3714	3716	3719	3721	3723	3726	3728	3730	3733	3735	3738	3814
3816	3818	3820	3822	3824	3826	3828	3920	3922	3924	3926	3928	3931	3933
3935	3937	3939	3941	3943	3945	3948	4030	4032	4034	4036	4038	4040	4042
4044	4046	4048	4050	4052	4055	4136	4138	4140	4142	4144	4147	4149	4151
4238	4240	4242	4244	4246	4251	4253	4255	4257	4259	4262	4343	4345	4347
4349	4351	4353	4355	4357	4359	4361	4363	4366	4368	4370	4449	4451	4453
4455	4457	4459	4461	4463	4465	4467	4470	4473	4552	4555	4557	4559	4561
4563	4565	4567	4569	4572	4574	4576	4578	4580	4582	4584	4586	4589	4591
4593	4595	4597	4599	4601	4603	4606	4608	4610	4612	4614	4616	4618	4620
4623	4625	4627	4629	4631	4633	4635	4637	4640	4642	4644	4646	4648	4650
4652	4654	4657	4659	4661	4663	4665	4667	4669	4671	4674	4676	4678	4680
4682	4684	4686	4688	4691	4693	4695	4697	4699	4701	4703	4705	4708	4710
4712	4714	4716	4718	4720	4722	4725	4727	4729	4731	4733	4735	4737	4739
4742	4744	4746	4748	4750	4752	4754	4756	4759	4761	4763	4765	4767	4769
4771	4773	4775	4853	4855	4857	4859	4861	4863	4865	4867	4869	4871	4873
4875	4970	4972	4974	4976	4978	4980	4982	4984	4986	4989	4991	4993	4995
4997	4999	5001	5003	5005	5008	5010	5012	5014	5016	5018	5020	5022	5024
5027	5029	5031	5033	5035	5037	5039	5041	5043	5046	5048	5050	5052	5054
5056	5058	5060	5063	5174	5177	5180	5183	5194	5196	5199	5201	5203	5206
5208	5210	5213	5215	5217	5220	5222	5224	5227	5229	5231	5234	5236	5238
5241	5243	5245	5248	5250	5252	5255	5257	5259	5262	5264	5266	5269	5271
5273	5276	5278	5280	5283	5285	5287	5290	5292	5294	5297	5299	5301	5304
5306	5308	5311	5313	5315	5318	5320	5322	5325	5327	5329	5332	5334	5336
5339	5341	5343	5346	5348	5350	5353	5355	5357	5360	5362	5364	5367	5369
5371	5374	5376	5378	5381	5383	5386	5388	5390	5392	5394	5396	5398	5400
5402	5404	5406	5408	5410	5412	5414	5416	5418	5420	5422	5424	5426	5428
5430	5432	5434	5436	5521	5523	5525	5527	5529	5531	5533	5535	5537	5539
5542	5544	5546	5548	5550	5552	5554	5634	5636	5638	5640	5642	5644	5646
5648	5650	5652	5655	5657	5659	5661	5663	5665	5667	5669	5671	5673	5675
5677	5679	5681	5683	5685	5687	5689	5691	5693	5695	5697	5699	5701	5703
5705	5707	5709	5711	5713	5715	5717	5719	5721	5723	5725	5727	5729	5731
5733	5735	5737	5739	5741	5743	5745	5747	5749	5752	5754	5757	5759	5761
5763	5765	5767	5769	5771	5773	5775	5777	5779	5781	5783	5785	5787	5789

SEQ 0897

	5791	5793	5795	5797	5799	5801	5803	5880	5882	5834	5886	5888	5890	5892
	5894	5896	5898	5901	5903	5980	5982	5984	5986	5988	6066	6068	6070	6072
	6074	6076	6079	6160	6162	6164	6166	6168	6170	6172	6174	6176	6179	6181
	6183	6185	6188	6265	6267	6269	6271	6273	6275	6277	6279	6281	6283	6285
	6287	6289	6291	6293	6295	6373	6375	6377	6379	6455	6457	6459	6461	6463
	6465	6467	6469	6471	6473	6475	6477	6479	6481	6483	6485	6487	6489	6491
	6493	6495	6497	6499	6501	6503	6505	6507	6509	6511	6513	6515	6517	6519
	6521	6524	6527	6602	6604	6606	6608	6610	6612	6614	6616	6618	6620	6622
	6624	6626	6628	6630	6632	6634	6636	6638	6640	6642	6644	6646	6648	6650
	6652	6654	6656	6658	6660	6662	6664	6666	6669	6672	6746	6748	6750	6752
	6754	6756	6758	6760	6762	6764	6766	6768	6770	6772	6774	6776	6778	6780
	6782	6784	6786	6788	6790	6792	6794	6796	6798	6800	6802	6804	6806	6808
	6810	6812	6814	6816	6818	6820	6822	6824	6826	6828	6830	6832	6834	6836
	6838	6840	6842	6845	6848	6921	6923	6925	6927	6929	6931	6933	6935	6937
	6939	6941	6943	6945	6947	6949	6951	6953	6955	6957	6959	6961	6963	6965
	6967	6969	6971	6973	6975	6977	6979	6981	6983	6985	6987	6989	6991	6993
	6995	6998	7000	7002										
FIELD	194	196	198	200	298	300	302	304	306	309	311	313	315	409
	411	413	415	417	419	421	423	425	427	429	431	433	436	438
	532	534	536	538	540	542	544	546	548	550	552	554	556	559
	561	665	667	669	671	673	675	677	679	681	683	685	687	689
	691	693	695	697	699	701	703	705	707	709	711	797	799	801
	803	888	890	892	894	979	981	983	985	1067	1069	1071	1149	1151
	1153	1155	1157	1159	1161	1164	1240	1242	1244	1246	1327	1329	1331	1333
	1335	1337	1339	1341	1343	1345	1347	1349	1351	1353	1356	1358	1360	1363
	1439	1441	1443	1445	1447	1449	1451	1453	1455	1457	1459	1461	1463	1465
	1467	1469	1471	1473	1475	1477	1479	1481	1483	1485	1487	1489	1491	1493
	1495	1497	1499	1501	1503	1505	1507	1509	1511	1513	1515	1517	1519	1521
	1523	1526	1528	1604	1606	1608	1610	1612	1614	1616	1618	1620	1622	1624
	1626	1628	1630	1632	1634	1636	1638	1640	1642	1644	1646	1648	1650	1652
	1654	1656	1658	1660	1662	1664	1666	1668	1670	1672	1674	1676	1678	1680
	1682	1684	1686	1688	1691	1693	1769	1771	1773	1775	1777	1779	1781	1783
	1785	1787	1789	1791	1793	1795	1797	1799	1801	1803	1805	1807	1809	1811
	1813	1815	1817	1819	1821	1823	1825	1827	1829	1831	1833	1835	1837	1839
	1841	1843	1845	1847	1849	1851	1853	1855	1858	1860	1936	1938	1940	1942
	1944	1946	1948	1950	1952	1954	1956	1958	1960	1962	1964	1966	1968	1970
	1972	1974	1976	1978	1980	1982	1984	1986	1988	1990	1992	1994	1996	1998
	2000	2002	2004	2006	2008	2010	2012	2014	2016	2018	2021	2023	2099	2101
	2103	2105	2107	2109	2111	2113	2115	2117	2119	2121	2123	2125	2127	2129
	2131	2133	2135	2137	2139	2141	2143	2145	2147	2149	2151	2153	2155	2157
	2159	2161	2163	2165	2167	2169	2171	2173	2175	2177	2179	2181	2183	2185
	2188	2190	2269	2271	2273	2275	2277	2279	2281	2361	2363	2365	2367	2369
	2371	2373	2375	2454	2456	2458	2460	2462	2464	2466	2468	2470	2549	2551
	2553	2555	2557	2559	2561	2563	2565	2567	2646	2648	2650	2652	2654	2656
	2658	2660	2662	2664	2666	2752	2754	2756	2758	2760	2762	2764	2766	2769
	2771	2773	2775	2854	2856	2858	2860	2862	2864	2866	2869	2871	2873	2875
	2877	2879	2881	2884	2886	2888	2890	2892	2894	2896	2899	2977	2979	2981
	2983	2985	2987	2989	3064	3066	3152	3154	3156	3158	3160	3162	3164	3166
	3168	3170	3252	3254	3256	3258	3260	3263	3265	3267	3269	3272	3274	3276
	3279	3281	3283	3286	3288	3290	3293	3295	3297	3300	3302	3304	3307	3309
	3311	3314	3316	3318	3321	3323	3325	3328	3330	3332	3335	3337	3339	3342
	3344	3346	3349	3351	3353	3356	3358	3360	3363	3365	3367	3370	3372	3374
	3377	3379	3381	3384	3386	3388	3391	3393	3395	3398	3400	3480	3482	3484

SEQ 0910

```
            3486   3488   3490   3492   3494   3496   3498   3500   3503   3505   3589   3591   3593   3595
            3598   3600   3602   3605   3607   3609   3612   3614   3616   3619   3621   3702   3705   3707        SEQ 0911
            3709   3712   3714   3716   3719   3721   3723   3726   3728   3730   3733   3735   3738   3814
            3816   3818   3820   3822   3824   3826   3828   3920   3922   3924   3926   3928   3931   3933
            3935   3937   3939   3941   3943   3945   3948   4030   4032   4034   4036   4038   4040   4042
            4044   4046   4048   4050   4052   4055   4136   4138   4140   4142   4144   4147   4149   4151
            4238   4240   4242   4244   4246   4251   4253   4255   4257   4259   4262   4343   4345   4347
            4349   4351   4353   4355   4357   4359   4361   4363   4366   4368   4370   4449   4451   4453
            4455   4457   4459   4461   4463   4465   4467   4470   4473   4552   4555   4557   4559   4561
            4563   4565   4567   4569   4572   4574   4576   4578   4580   4582   4584   4586   4589   4591
            4593   4595   4597   4599   4601   4603   4606   4608   4610   4612   4614   4616   4618   4620
            4623   4625   4627   4629   4631   4633   4635   4637   4640   4642   4644   4646   4648   4650
            4652   4654   4657   4659   4661   4663   4665   4667   4669   4671   4674   4676   4678   4680
            4682   4684   4686   4688   4691   4693   4695   4697   4699   4701   4703   4705   4708   4710
            4712   4714   4716   4718   4720   4722   4725   4727   4729   4731   4733   4735   4737   4739
            4742   4744   4746   4748   4750   4752   4754   4756   4759   4761   4763   4765   4767   4769
            4771   4773   4775   4853   4855   4857   4859   4861   4863   4865   4867   4869   4871   4873
            4875   4970   4972   4974   4976   4978   4980   4982   4984   4986   4989   4991   4993   4995
            4997   4999   5001   5003   5005   5008   5010   5012   5014   5016   5018   5020   5022   5024
            5027   5029   5031   5033   5035   5037   5039   5041   5043   5046   5048   5050   5052   5054
            5056   5058   5060   5063   5174   5177   5180   5183   5194   5196   5199   5201   5203   5206
            5208   5210   5213   5215   5217   5220   5222   5224   5227   5229   5231   5234   5236   5238
            5241   5243   5245   5248   5250   5252   5255   5257   5259   5262   5264   5266   5269   5271
            5273   5276   5278   5280   5283   5285   5287   5290   5292   5294   5297   5299   5301   5304
            5306   5308   5311   5313   5315   5318   5320   5322   5325   5327   5329   5332   5334   5336
            5339   5341   5343   5346   5348   5350   5353   5355   5357   5360   5362   5364   5367   5369
            5371   5374   5376   5378   5381   5383   5386   5388   5390   5392   5394   5396   5398   5400
            5402   5404   5406   5408   5410   5412   5414   5416   5418   5420   5422   5424   5426   5428
            5430   5432   5434   5436   5521   5523   5525   5527   5529   5531   5533   5535   5537   5539
            5542   5544   5546   5548   5550   5552   5554   5634   5636   5638   5640   5642   5644   5646
            5648   5650   5652   5655   5657   5659   5661   5663   5665   5667   5669   5671   5673   5675
            5677   5679   5681   5683   5685   5687   5689   5691   5693   5695   5697   5699   5701   5703
            5705   5707   5709   5711   5713   5715   5717   5719   5721   5723   5725   5727   5729   5731
            5733   5735   5737   5739   5741   5743   5745   5747   5749   5752   5754   5757   5759   5761
            5763   5765   5767   5769   5771   5773   5775   5777   5779   5781   5783   5785   5787   5789
            5791   5793   5795   5797   5799   5801   5803   5880   5882   5884   5886   5888   5890   5892
            5894   5896   5898   5901   5903   5980   5982   5984   5986   5988   6066   6068   6070   6072
            6074   6076   6079   6160   6162   6164   6166   6168   6170   6172   6174   6176   6179   6181
            6183   6185   6188   6265   6267   6269   6271   6273   6275   6277   6279   6281   6283   6285
            6287   6289   6291   6293   6295   6373   6375   6377   6379   6455   6457   6459   6461   6463
            6465   6467   6469   6471   6473   6475   6477   6479   6481   6483   6485   6487   6489   6491
            6493   6495   6497   6499   6501   6503   6505   6507   6509   6511   6513   6515   6517   6519
            6521   6524   6527   6602   6604   6606   6608   6610   6612   6614   6616   6618   6620   6622
            6624   6626   6628   6630   6632   6634   6636   6638   6640   6642   6644   6646   6648   6650
            6652   6654   6656   6658   6660   6662   6664   6666   6669   6672   6746   6748   6750   6752
            6754   6756   6758   6760   6762   6764   6766   6768   6770   6772   6774   6776   6778   6780
            6782   6784   6786   6788   6790   6792   6794   6796   6798   6800   6802   6804   6806   6808
            6810   6812   6814   6816   6818   6820   6822   6824   6826   6828   6830   6832   6834   6836
            6838   6840   6842   6845   6848   6921   6923   6925   6927   6929   6931   6933   6935   6937
            6939   6941   6943   6945   6947   6949   6951   6953   6955   6957   6959   6961   6963   6965
            6967   6969   6971   6973   6975   6977   6979   6981   6983   6985   6987   6989   6991   6993
            6995   6998   7000   7002
GET         7171
```

```
GO         86     90     98    182    253    255    260    267    364    366    371    378    487    489        SEQ 0912
          494    501    611    613    618    625    752    754    759    766    843    845    850    857
          934    936    941    948   1025   1027   1032   1039   1107   1109   1114   1121   1198   1200
         1205   1212   1285   1287   1292   1299   1397   1399   1404   1411   1562   1564   1569   1576
         1727   1729   1734   1741   1894   1896   1901   1908   2057   2059   2064   2071   2227   2229
         2234   2241   2319   2321   2326   2333   2412   2414   2419   2426   2507   2509   2514   2521
         2604   2606   2611   2618   2710   2712   2717   2724   2812   2814   2819   2826   2935   2937
         2942   2949   3022   3024   3029   3036   3110   3112   3117   3124   3210   3212   3217   3224
         3438   3440   3445   3452   3547   3549   3554   3561   3660   3662   3667   3674   3772   3774
         3779   3786   3875   3877   3882   3889   3988   3990   3995   4002   4094   4096   4101   4108
         4193   4195   4200   4207   4301   4303   4308   4315   4407   4409   4414   4421   4510   4512
         4517   4524   4811   4813   4818   4825   4925   4927   4932   4939   5119   5121   5126   5134
         5168   5479   5481   5486   5493   5590   5592   5597   5604   5838   5840   5845   5852   5938
         5940   5945   5952   6024   6026   6031   6038   6118   6120   6125   6132   6223   6225   6230
         6237   6331   6333   6338   6345   6413   6415   6420   6427   6560   6562   6567   6574   6704
         6706   6711   6718   6879   6881   6886   6893   7041   7045   7046   7053   7058   7072   7073
         7090   7096   7105   7106   7108   7140   7157   7168   7170
MFLD     194#    194   196#    196   198#    198   200#    200   298#    298   300#    300   302#    302
         304#    304   306#    306   309#    309   311#    311   313#    313   315#    315   409#    409
         411#    411   413#    413   415#    415   417#    417   419#    419   421#    421   423#    423
         425#    425   427#    427   429#    429   431#    431   433#    433   436#    436   438#    438
         532#    532   534#    534   536#    536   538#    538   540#    540   542#    542   544#    544
         546#    546   548#    548   550#    550   552#    552   554#    554   556#    556   559#    559
         561#    561   665#    665   667#    667   669#    669   671#    671   673#    673   675#    675
         677#    677   679#    679   681#    681   683#    683   685#    685   687#    687   689#    689
         691#    691   693#    693   695#    695   697#    697   699#    699   701#    701   703#    703
         705#    705   707#    707   709#    709   711#    711   797#    797   799#    799   801#    801
         803#    803   888#    888   890#    890   892#    892   894#    894   979#    979   981#    981
         983#    983   985#    985  1067#   1067  1069#   1069  1071#   1071  1149#   1149  1151#   1151
        1153#   1153  1155#   1155  1157#   1157  1159#   1159  1161#   1161  1164#   1164  1240#   1240
        1242#   1242  1244#   1244  1246#   1246  1327#   1327  1329#   1329  1331#   1331  1333#   1333
        1335#   1335  1337#   1337  1339#   1339  1341#   1341  1343#   1343  1345#   1345  1347#   1347
        1349#   1349  1351#   1351  1353#   1353  1356#   1356  1358#   1358  1360#   1360  1363#   1363
        1439#   1439  1441#   1441  1443#   1443  1445#   1445  1447#   1447  1449#   1449  1451#   1451
        1453#   1453  1455#   1455  1457#   1457  1459#   1459  1461#   1461  1463#   1463  1465#   1465
        1467#   1467  1469#   1469  1471#   1471  1473#   1473  1475#   1475  1477#   1477  1479#   1479
        1481#   1481  1483#   1483  1485#   1485  1487#   1487  1489#   1489  1491#   1491  1493#   1493
        1495#   1495  1497#   1497  1499#   1499  1501#   1501  1503#   1503  1505#   1505  1507#   1507
        1509#   1509  1511#   1511  1513#   1513  1515#   1515  1517#   1517  1519#   1519  1521#   1521
        1523#   1523  1526#   1526  1528#   1528  1604#   1604  1606#   1606  1608#   1608  1610#   1610
        1612#   1612  1614#   1614  1616#   1616  1618#   1618  1620#   1620  1622#   1622  1624#   1624
        1626#   1626  1628#   1628  1630#   1630  1632#   1632  1634#   1634  1636#   1636  1638#   1638
        1640#   1640  1642#   1642  1644#   1644  1646#   1646  1648#   1648  1650#   1650  1652#   1652
        1654#   1654  1656#   1656  1658#   1658  1660#   1660  1662#   1662  1664#   1664  1666#   1666
        1668#   1668  1670#   1670  1672#   1672  1674#   1674  1676#   1676  1678#   1678  1680#   1680
        1682#   1682  1684#   1684  1686#   1686  1688#   1688  1691#   1691  1693#   1693  1769#   1769
        1771#   1771  1773#   1773  1775#   1775  1777#   1777  1779#   1779  1781#   1781  1783#   1783
        1785#   1785  1787#   1787  1789#   1789  1791#   1791  1793#   1793  1795#   1795  1797#   1797
        1799#   1799  1801#   1801  1803#   1803  1805#   1805  1807#   1807  1809#   1809  1811#   1811
        1813#   1813  1815#   1815  1817#   1817  1819#   1819  1821#   1821  1823#   1823  1825#   1825
        1827#   1827  1829#   1829  1831#   1831  1833#   1833  1835#   1835  1837#   1837  1839#   1839
        1841#   1841  1843#   1843  1845#   1845  1847#   1847  1849#   1849  1851#   1851  1853#   1853
        1855#   1855  1858#   1858  1860#   1860  1936#   1936  1938#   1938  1940#   1940  1942#   1942
```

```
1944#  1944  1946#  1946  1948#  1948  1950#  1950  1952#  1952  1954#  1954  1956#  1956     SEQ 0913
1958#  1958  1960#  1960  1962#  1962  1964#  1964  1966#  1966  1968#  1968  1970#  1970
1972#  1972  1974#  1974  1976#  1976  1978#  1978  1980#  1980  1982#  1982  1984#  1984
1986#  1986  1988#  1988  1990#  1990  1992#  1992  1994#  1994  1996#  1996  1998#  1998
2000#  2000  2002#  2002  2004#  2004  2006#  2006  2008#  2008  2010#  2010  2012#  2012
2014#  2014  2016#  2016  2018#  2018  2021#  2021  2023#  2023  2099#  2099  2101#  2101
2103#  2103  2105#  2105  2107#  2107  2109#  2109  2111#  2111  2113#  2113  2115#  2115
2117#  2117  2119#  2119  2121#  2121  2123#  2123  2125#  2125  2127#  2127  2129#  2129
2131#  2131  2133#  2133  2135#  2135  2137#  2137  2139#  2139  2141#  2141  2143#  2143
2145#  2145  2147#  2147  2149#  2149  2151#  2151  2153#  2153  2155#  2155  2157#  2157
2159#  2159  2161#  2161  2163#  2163  2165#  2165  2167#  2167  2169#  2169  2171#  2171
2173#  2173  2175#  2175  2177#  2177  2179#  2179  2181#  2181  2183#  2183  2185#  2185
2188#  2188  2190#  2190  2269#  2269  2271#  2271  2273#  2273  2275#  2275  2277#  2277
2279#  2279  2281#  2281  2361#  2361  2363#  2363  2365#  2365  2367#  2367  2369#  2369
2371#  2371  2373#  2373  2375#  2375  2454#  2454  2456#  2456  2458#  2458  2460#  2460
2462#  2462  2464#  2464  2466#  2466  2468#  2468  2470#  2470  2549#  2549  2551#  2551
2553#  2553  2555#  2555  2557#  2557  2559#  2559  2561#  2561  2563#  2563  2565#  2565
2567#  2567  2646#  2646  2648#  2648  2650#  2650  2652#  2652  2654#  2654  2656#  2656
2658#  2658  2660#  2660  2662#  2662  2664#  2664  2666#  2666  2752#  2752  2754#  2754
2756#  2756  2758#  2758  2760#  2760  2762#  2762  2764#  2764  2766#  2766  2769#  2769
2771#  2771  2773#  2773  2775#  2775  2854#  2854  2856#  2856  2858#  2858  2860#  2860
2862#  2862  2864#  2864  2866#  2866  2869#  2869  2871#  2871  2873#  2873  2875#  2875
2877#  2877  2879#  2879  2881#  2881  2884#  2884  2886#  2886  2888#  2888  2890#  2890
2892#  2892  2894#  2894  2896#  2896  2899#  2899  2977#  2977  2979#  2979  2981#  2981
2983#  2983  2985#  2985  2987#  2987  2989#  2989  3064#  3064  3066#  3066  3152#  3152
3154#  3154  3156#  3156  3158#  3158  3160#  3160  3162#  3162  3164#  3164  3166#  3166
3168#  3168  3170#  3170  3252#  3252  3254#  3254  3256#  3256  3258#  3258  3260#  3260
3263#  3263  3265#  3265  3267#  3267  3269#  3269  3272#  3272  3274#  3274  3276#  3276
3279#  3279  3281#  3281  3283#  3283  3286#  3286  3288#  3288  3290#  3290  3293#  3293
3295#  3295  3297#  3297  3300#  3300  3302#  3302  3304#  3304  3307#  3307  3309#  3309
3311#  3311  3314#  3314  3316#  3316  3318#  3318  3321#  3321  3323#  3323  3325#  3325
3328#  3328  3330#  3330  3332#  3332  3335#  3335  3337#  3337  3339#  3339  3342#  3342
3344#  3344  3346#  3346  3349#  3349  3351#  3351  3353#  3353  3356#  3356  3358#  3358
3360#  3360  3363#  3363  3365#  3365  3367#  3367  3370#  3370  3372#  3372  3374#  3374
3377#  3377  3379#  3379  3381#  3381  3384#  3384  3386#  3386  3388#  3388  3391#  3391
3393#  3393  3395#  3395  3398#  3398  3400#  3400  3480#  3480  3482#  3482  3484#  3484
3486#  3486  3488#  3488  3490#  3490  3492#  3492  3494#  3494  3496#  3496  3498#  3498
3500#  3500  3503#  3503  3505#  3505  3589#  3589  3591#  3591  3593#  3593  3595#  3595
3598#  3598  3600#  3600  3602#  3602  3605#  3605  3607#  3607  3609#  3609  3612#  3612
3614#  3614  3616#  3616  3619#  3619  3621#  3621  3702#  3702  3705#  3705  3707#  3707
3709#  3709  3712#  3712  3714#  3714  3716#  3716  3719#  3719  3721#  3721  3723#  3723
3726#  3726  3728#  3728  3730#  3730  3733#  3733  3735#  3735  3738#  3738  3814#  3814
3816#  3816  3818#  3818  3820#  3820  3822#  3822  3824#  3824  3826#  3826  3828#  3828
3920#  3920  3922#  3922  3924#  3924  3926#  3926  3928#  3928  3931#  3931  3933#  3933
3935#  3935  3937#  3937  3939#  3939  3941#  3941  3943#  3943  3945#  3945  3948#  3948
4030#  4030  4032#  4032  4034#  4034  4036#  4036  4038#  4038  4040#  4040  4042#  4042
4044#  4044  4046#  4046  4048#  4048  4050#  4050  4052#  4052  4055#  4055  4136#  4136
4138#  4138  4140#  4140  4142#  4142  4144#  4144  4147#  4147  4149#  4149  4151#  4151
4238#  4238  4240#  4240  4242#  4242  4244#  4244  4246#  4246  4251#  4251  4253#  4253
4255#  4255  4257#  4257  4259#  4259  4262#  4262  4343#  4343  4345#  4345  4347#  4347
4349#  4349  4351#  4351  4353#  4353  4355#  4355  4357#  4357  4359#  4359  4361#  4361
4363#  4363  4366#  4366  4368#  4368  4370#  4370  4449#  4449  4451#  4451  4453#  4453
4455#  4455  4457#  4457  4459#  4459  4461#  4461  4463#  4463  4465#  4465  4467#  4467
```

```
4470#  4470  4473#  4473  4552#  4552  4555#  4555  4557#  4557  4559#  4559  4561#  4561      SEQ 0914
4563#  4563  4565#  4565  4567#  4567  4569#  4569  4572#  4572  4574#  4574  4576#  4576
4578#  4578  4580#  4580  4582#  4582  4584#  4584  4586#  4586  4589#  4589  4591#  4591
4593#  4593  4595#  4595  4597#  4597  4599#  4599  4601#  4601  4603#  4603  4606#  4606
4608#  4608  4610#  4610  4612#  4612  4614#  4614  4616#  4616  4618#  4618  4620#  4620
4623#  4623  4625#  4625  4627#  4627  4629#  4629  4631#  4631  4633#  4633  4635#  4635
4637#  4637  4640#  4640  4642#  4642  4644#  4644  4646#  4646  4648#  4648  4650#  4650
4652#  4652  4654#  4654  4657#  4657  4659#  4659  4661#  4661  4663#  4663  4665#  4665
4667#  4667  4669#  4669  4671#  4671  4674#  4674  4676#  4676  4678#  4678  4680#  4680
4682#  4682  4684#  4684  4686#  4686  4688#  4688  4691#  4691  4693#  4693  4695#  4695
4697#  4697  4699#  4699  4701#  4701  4703#  4703  4705#  4705  4708#  4708  4710#  4710
4712#  4712  4714#  4714  4716#  4716  4718#  4718  4720#  4720  4722#  4722  4725#  4725
4727#  4727  4729#  4729  4731#  4731  4733#  4733  4735#  4735  4737#  4737  4739#  4739
4742#  4742  4744#  4744  4746#  4746  4748#  4748  4750#  4750  4752#  4752  4754#  4754
4756#  4756  4759#  4759  4761#  4761  4763#  4763  4765#  4765  4767#  4767  4769#  4769
4771#  4771  4773#  4773  4775#  4775  4853#  4853  4855#  4855  4857#  4857  4859#  4859
4861#  4861  4863#  4863  4865#  4865  4867#  4867  4869#  4869  4871#  4871  4873#  4873
4875#  4875  4970#  4970  4972#  4972  4974#  4974  4976#  4976  4978#  4978  4980#  4980
4982#  4982  4984#  4984  4986#  4986  4989#  4989  4991#  4991  4993#  4993  4995#  4995
4997#  4997  4999#  4999  5001#  5001  5003#  5003  5005#  5005  5008#  5008  5010#  5010
5012#  5012  5014#  5014  5016#  5016  5018#  5018  5020#  5020  5022#  5022  5024#  5024
5027#  5027  5029#  5029  5031#  5031  5033#  5033  5035#  5035  5037#  5037  5039#  5039
5041#  5041  5043#  5043  5046#  5046  5048#  5048  5050#  5050  5052#  5052  5054#  5054
5056#  5056  5058#  5058  5060#  5060  5063#  5063  5174#  5174  5177#  5177  5180#  5180
5183#  5183  5194#  5194  5196#  5196  5199#  5199  5201#  5201  5203#  5203  5206#  5206
5208#  5208  5210#  5210  5213#  5213  5215#  5215  5217#  5217  5220#  5220  5222#  5222
5224#  5224  5227#  5227  5229#  5229  5231#  5231  5234#  5234  5236#  5236  5238#  5238
5241#  5241  5243#  5243  5245#  5245  5248#  5248  5250#  5250  5252#  5252  5255#  5255
5257#  5257  5259#  5259  5262#  5262  5264#  5264  5266#  5266  5269#  5269  5271#  5271
5273#  5273  5276#  5276  5278#  5278  5280#  5280  5283#  5283  5285#  5285  5287#  5287
5290#  5290  5292#  5292  5294#  5294  5297#  5297  5299#  5299  5301#  5301  5304#  5304
5306#  5306  5308#  5308  5311#  5311  5313#  5313  5315#  5315  5318#  5318  5320#  5320
5322#  5322  5325#  5325  5327#  5327  5329#  5329  5332#  5332  5334#  5334  5336#  5336
5339#  5339  5341#  5341  5343#  5343  5346#  5346  5348#  5348  5350#  5350  5353#  5353
5355#  5355  5357#  5357  5360#  5360  5362#  5362  5364#  5364  5367#  5367  5369#  5369
5371#  5371  5374#  5374  5376#  5376  5378#  5378  5381#  5381  5383#  5383  5386#  5386
5388#  5388  5390#  5390  5392#  5392  5394#  5394  5396#  5396  5398#  5398  5400#  5400
5402#  5402  5404#  5404  5406#  5406  5408#  5408  5410#  5410  5412#  5412  5414#  5414
5416#  5416  5418#  5418  5420#  5420  5422#  5422  5424#  5424  5426#  5426  5428#  5428
5430#  5430  5432#  5432  5434#  5434  5436#  5436  5521#  5521  5523#  5523  5525#  5525
5527#  5527  5529#  5529  5531#  5531  5533#  5533  5535#  5535  5537#  5537  5539#  5539
5542#  5542  5544#  5544  5546#  5546  5548#  5548  5550#  5550  5552#  5552  5554#  5554
5634#  5634  5636#  5636  5638#  5638  5640#  5640  5642#  5642  5644#  5644  5646#  5646
5648#  5648  5650#  5650  5652#  5652  5655#  5655  5657#  5657  5659#  5659  5661#  5661
5663#  5663  5665#  5665  5667#  5667  5669#  5669  5671#  5671  5673#  5673  5675#  5675
5677#  5677  5679#  5679  5681#  5681  5683#  5683  5685#  5685  5687#  5687  5689#  5689
5691#  5691  5693#  5693  5695#  5695  5697#  5697  5699#  5699  5701#  5701  5703#  5703
5705#  5705  5707#  5707  5709#  5709  5711#  5711  5713#  5713  5715#  5715  5717#  5717
5719#  5719  5721#  5721  5723#  5723  5725#  5725  5727#  5727  5729#  5729  5731#  5731
5733#  5733  5735#  5735  5737#  5737  5739#  5739  5741#  5741  5743#  5743  5745#  5745
5747#  5747  5749#  5749  5752#  5752  5754#  5754  5757#  5757  5759#  5759  5761#  5761
5763#  5763  5765#  5765  5767#  5767  5769#  5769  5771#  5771  5773#  5773  5775#  5775
5777#  5777  5779#  5779  5781#  5781  5783#  5783  5785#  5785  5787#  5787  5789#  5789
```

```
0902
              5791#  5791   5793#  5793   5795#  5795   5797#  5797   5799#  5799   5801#  5801   5803#  5803     SEQ 0915
              5880#  5880   5882#  5882   5884#  5884   5886#  5886   5888#  5888   5890#  5890   5892#  5892
              5894#  5894   5896#  5896   5898#  5898   5901#  5901   5903#  5903   5980#  5980   5982#  5982
              5984#  5984   5986#  5986   5988#  5988   6066#  6066   6068#  6068   6070#  6070   6072#  6072
              6074#  6074   6076#  6076   6079#  6079   6160#  6160   6162#  6162   6164#  6164   6166#  6166
              6168#  6168   6170#  6170   6172#  6172   6174#  6174   6176#  6176   6179#  6179   6181#  6181
              6183#  6183   6185#  6185   6188#  6188   6265#  6265   6267#  6267   6269#  6269   6271#  6271
              6273#  6273   6275#  6275   6277#  6277   6279#  6279   6281#  6281   6283#  6283   6285#  6285
              6287#  6287   6289#  6289   6291#  6291   6293#  6293   6295#  6295   6373#  6373   6375#  6375
              6377#  6377   6379#  6379   6455#  6455   6457#  6457   6459#  6459   6461#  6461   6463#  6463
              6465#  6465   6467#  6467   6469#  6469   6471#  6471   6473#  6473   6475#  6475   6477#  6477
              6479#  6479   6481#  6481   6483#  6483   6485#  6485   6487#  6487   6489#  6489   6491#  6491
              6493#  6493   6495#  6495   6497#  6497   6499#  6499   6501#  6501   6503#  6503   6505#  6505
              6507#  6507   6509#  6509   6511#  6511   6513#  6513   6515#  6515   6517#  6517   6519#  6519
              6521#  6521   6524#  6524   6527#  6527   6602#  6602   6604#  6604   6606#  6606   6608#  6608
              6610#  6610   6612#  6612   6614#  6614   6616#  6616   6618#  6618   6620#  6620   6622#  6622
              6624#  6624   6626#  6626   6628#  6628   6630#  6630   6632#  6632   6634#  6634   6636#  6636
              6638#  6638   6640#  6640   6642#  6642   6644#  6644   6646#  6646   6648#  6648   6650#  6650
              6652#  6652   6654#  6654   6656#  6656   6658#  6658   6660#  6660   6662#  6662   6664#  6664
              6666#  6666   6669#  6669   6672#  6672   6746#  6746   6748#  6748   6750#  6750   6752#  6752
              6754#  6754   6756#  6756   6758#  6758   6760#  6760   6762#  6762   6764#  6764   6766#  6766
              6768#  6768   6770#  6770   6772#  6772   6774#  6774   6776#  6776   6778#  6778   6780#  6780
              6782#  6782   6784#  6784   6786#  6786   6788#  6788   6790#  6790   6792#  6792   6794#  6794
              6796#  6796   6798#  6798   6800#  6800   6802#  6802   6804#  6804   6806#  6806   6808#  6808
              6810#  6810   6812#  6812   6814#  6814   6816#  6816   6818#  6818   6820#  6820   6822#  6822
              6824#  6824   6826#  6826   6828#  6828   6830#  6830   6832#  6832   6834#  6834   6836#  6836
              6838#  6838   6840#  6840   6842#  6842   6845#  6845   6848#  6848   6921#  6921   6923#  6923
              6925#  6925   6927#  6927   6929#  6929   6931#  6931   6933#  6933   6935#  6935   6937#  6937
              6939#  6939   6941#  6941   6943#  6943   6945#  6945   6947#  6947   6949#  6949   6951#  6951
              6953#  6953   6955#  6955   6957#  6957   6959#  6959   6961#  6961   6963#  6963   6965#  6965
              6967#  6967   6969#  6969   6971#  6971   6973#  6973   6975#  6975   6977#  6977   6979#  6979
              6981#  6981   6983#  6983   6985#  6985   6987#  6987   6989#  6989   6991#  6991   6993#  6993
              6995#  6995   6998#  6998   7000#  7000   7002#  7002
MWORD          194    196    198    200    298    300    302    304     306    309    311    313    315    409
               411    413    415    417    419    421    423    425     427    429    431    433    436    438
               532    534    536    538    540    542    544    546     548    550    552    554    556    559
               561    665    667    669    671    673    675    677     679    681    683    685    687    689
               691    693    695    697    699    701    703    705     707    709    711    797    799    801
               803    888    890    892    894    979    981    983     985   1067   1069   1071   1149   1151
              1153   1155   1157   1159   1161   1164   1240   1242    1244   1246   1327   1329   1331   1333
              1335   1337   1339   1341   1343   1345   1347   1349    1351   1353   1356   1358   1360   1363
              1439   1441   1443   1445   1447   1449   1451   1453    1455   1457   1459   1461   1463   1465
              1467   1469   1471   1473   1475   1477   1479   1481    1483   1485   1487   1489   1491   1493
              1495   1497   1499   1501   1503   1505   1507   1509    1511   1513   1515   1517   1519   1521
              1523   1526   1528   1604   1606   1608   1610   1612    1614   1616   1618   1620   1622   1624
              1626   1628   1630   1632   1634   1636   1638   1640    1642   1644   1646   1648   1650   1652
              1654   1656   1658   1660   1662   1664   1666   1668    1670   1672   1674   1676   1678   1680
              1682   1684   1686   1688   1691   1693   1769   1771    1773   1775   1777   1779   1781   1783
              1785   1787   1789   1791   1793   1795   1797   1799    1801   1803   1805   1807   1809   1811
              1813   1815   1817   1819   1821   1823   1825   1827    1829   1831   1833   1835   1837   1839
              1841   1843   1845   1847   1849   1851   1853   1855    1858   1860   1936   1938   1940   1942
              1944   1946   1948   1950   1952   1954   1956   1958    1960   1962   1964   1966   1968   1970
              1972   1974   1976   1978   1980   1982   1984   1986    1988   1990   1992   1994   1996   1998
```

0903

```
2000  2002  2004  2006  2008  2010  2012  2014  2016  2018  2021  2023  2099  2101
2103  2105  2107  2109  2111  2113  2115  2117  2119  2121  2123  2125  2127  2129
2131  2133  2135  2137  2139  2141  2143  2145  2147  2149  2151  2153  2155  2157
2159  2161  2163  2165  2167  2169  2171  2173  2175  2177  2179  2181  2183  2185
2188  2190  2269  2271  2273  2275  2277  2279  2281  2361  2363  2365  2367  2369
2371  2373  2375  2454  2456  2458  2460  2462  2464  2466  2468  2470  2549  2551
2553  2555  2557  2559  2561  2563  2565  2567  2646  2648  2650  2652  2654  2656
2658  2660  2662  2664  2666  2752  2754  2756  2758  2760  2762  2764  2766  2769
2771  2773  2775  2854  2856  2858  2860  2862  2864  2866  2869  2871  2873  2875
2877  2879  2881  2884  2886  2888  2890  2892  2894  2896  2899  2977  2979  2981
2983  2985  2987  2989  3064  3066  3152  3154  3156  3158  3160  3162  3164  3166
3168  3170  3252  3254  3256  3258  3260  3263  3265  3267  3269  3272  3274  3276
3279  3281  3283  3286  3288  3290  3293  3295  3297  3300  3302  3304  3307  3309
3311  3314  3316  3318  3321  3323  3325  3328  3330  3332  3335  3337  3339  3342
3344  3346  3349  3351  3353  3356  3358  3360  3363  3365  3367  3370  3372  3374
3377  3379  3381  3384  3386  3388  3391  3393  3395  3398  3400  3480  3482  3484
3486  3488  3490  3492  3494  3496  3498  3500  3503  3505  3589  3591  3593  3595
3598  3600  3602  3605  3607  3609  3612  3614  3616  3619  3621  3702  3705  3707
3709  3712  3714  3716  3719  3721  3723  3726  3728  3730  3733  3735  3738  3814
3816  3818  3820  3822  3824  3826  3828  3920  3922  3924  3926  3928  3931  3933
3935  3937  3939  3941  3943  3945  3948  4030  4032  4034  4036  4038  4040  4042
4044  4046  4048  4050  4052  4055  4136  4138  4140  4142  4144  4147  4149  4151
4238  4240  4242  4244  4246  4251  4253  4255  4257  4259  4262  4343  4345  4347
4349  4351  4353  4355  4357  4359  4361  4363  4366  4368  4370  4449  4451  4453
4455  4457  4459  4461  4463  4465  4467  4470  4473  4552  4555  4557  4559  4561
4563  4565  4567  4569  4572  4574  4576  4578  4580  4582  4584  4586  4589  4591
4593  4595  4597  4599  4601  4603  4606  4608  4610  4612  4614  4616  4618  4620
4623  4625  4627  4629  4631  4633  4635  4637  4640  4642  4644  4646  4648  4650
4652  4654  4657  4659  4661  4663  4665  4667  4669  4671  4674  4676  4678  4680
4682  4684  4686  4688  4691  4693  4695  4697  4699  4701  4703  4705  4708  4710
4712  4714  4716  4718  4720  4722  4725  4727  4729  4731  4733  4735  4737  4739
4742  4744  4746  4748  4750  4752  4754  4756  4759  4761  4763  4765  4767  4769
4771  4773  4775  4853  4855  4857  4859  4861  4863  4865  4867  4869  4871  4873
4875  4970  4972  4974  4976  4978  4980  4982  4984  4986  4989  4991  4993  4995
4997  4999  5001  5003  5005  5008  5010  5012  5014  5016  5018  5020  5022  5024
5027  5029  5031  5033  5035  5037  5039  5041  5043  5046  5048  5050  5052  5054
5056  5058  5060  5063  5174  5177  5180  5183  5194  5196  5199  5201  5203  5206
5208  5210  5213  5215  5217  5220  5222  5224  5227  5229  5231  5234  5236  5238
5241  5243  5245  5248  5250  5252  5255  5257  5259  5262  5264  5266  5269  5271
5273  5276  5278  5280  5283  5285  5287  5290  5292  5294  5297  5299  5301  5304
5306  5308  5311  5313  5315  5318  5320  5322  5325  5327  5329  5332  5334  5336
5339  5341  5343  5346  5348  5350  5353  5355  5357  5360  5362  5364  5367  5369
5371  5374  5376  5378  5381  5383  5386  5388  5390  5392  5394  5396  5398  5400
5402  5404  5406  5408  5410  5412  5414  5416  5418  5420  5422  5424  5426  5428
5430  5432  5434  5436  5521  5523  5525  5527  5529  5531  5533  5535  5537  5539
5542  5544  5546  5548  5550  5552  5554  5634  5636  5638  5640  5642  5644  5646
5648  5650  5652  5655  5657  5659  5661  5663  5665  5667  5669  5671  5673  5675
5677  5679  5681  5683  5685  5687  5689  5691  5693  5695  5697  5699  5701  5703
5705  5707  5709  5711  5713  5715  5717  5719  5721  5723  5725  5727  5729  5731
5733  5735  5737  5739  5741  5743  5745  5747  5749  5752  5754  5757  5759  5761
5763  5765  5767  5769  5771  5773  5775  5777  5779  5781  5783  5785  5787  5789
5791  5793  5795  5797  5799  5801  5803  5880  5882  5884  5886  5888  5890  5892
5894  5896  5898  5901  5903  5980  5982  5984  5986  5988  6066  6068  6070  6072
```

```
            6074   6076   6079   6160   6162   6164   6166   6168   6170   6172   6174   6176   6179   6181
            6183   6185   6188   6265   6267   6269   6271   6273   6275   6277   6279   6281   6283   6285
            6287   6289   6291   6293   6295   6373   6375   6377   6379   6455   6457   6459   6461   6463
            6465   6467   6469   6471   6473   6475   6477   6479   6481   6483   6485   6487   6489   6491
            6493   6495   6497   6499   6501   6503   6505   6507   6509   6511   6513   6515   6517   6519
            6521   6524   6527   6602   6604   6606   6608   6610   6612   6614   6616   6618   6620   6622
            6624   6626   6628   6630   6632   6634   6636   6638   6640   6642   6644   6646   6648   6650
            6652   6654   6656   6658   6660   6662   6664   6666   6669   6672   6746   6748   6750   6752
            6754   6756   6758   6760   6762   6764   6766   6768   6770   6772   6774   6776   6778   6780
            6782   6784   6786   6788   6790   6792   6794   6796   6798   6800   6802   6804   6806   6808
            6810   6812   6814   6816   6818   6820   6822   6824   6826   6828   6830   6832   6834   6836
            6838   6840   6842   6845   6848   6921   6923   6925   6927   6929   6931   6933   6935   6937
            6939   6941   6943   6945   6947   6949   6951   6953   6955   6957   6959   6961   6963   6965
            6967   6969   6971   6973   6975   6977   6979   6981   6983   6985   6987   6989   6991   6993
            6995   6998   7000   7002
PNT4        7112
PNTMSG      7109
PUT         7124
RTN          112    132    185    256    281    367    392    490    515    614    639    755    780    846
             871    937    962   1028   1053   1110   1135   1201   1226   1288   1313   1400   1425   1565
            1590   1730   1755   1897   1922   2060   2085   2230   2255   2322   2347   2415   2440   2510
            2535   2607   2632   2674   2713   2738   2815   2840   2938   2963   3025   3050   3113   3138
            3213   3238   3441   3466   3550   3575   3663   3688   3775   3800   3878   3903   3991   4016
            4097   4122   4196   4221   4304   4329   4410   4435   4513   4538   4814   4839   4928   4953
            5122   5148   5170   5482   5507   5593   5618   5841   5866   5941   5966   6027   6052   6121
            6146   6226   6251   6334   6359   6416   6441   6563   6588   6707   6732   6882   6907   7083
            7097   7113   7123
SCOPER       105    274    385    508    632    773    864    955   1046   1128   1219   1306   1418   1583
            1748   1915   2078   2248   2340   2433   2528   2625   2731   2833   2956   3043   3131   3231
            3459   3568   3681   3793   3896   4009   4115   4214   4322   4428   4531   4832   4946   5141
            5500   5611   5859   5959   6045   6139   6244   6352   6434   6581   6725   6900   7062   7077
STABLE       116    117    118    119    120    121    122    123    285    286    287    288    289    396
             397    398    399    400    519    520    521    522    523    643    644    645    646    647
             648    649    650    651    652    653    654    655    656    784    785    786    787    788
             875    876    877    878    879    966    967    968    969    970   1057   1058   1139   1140
            1230   1231   1317   1318   1429   1430   1594   1595   1759   1760   1926   1927   2089   2090
            2259   2260   2351   2352   2444   2445   2539   2540   2636   2637   2742   2743   2844   2845
            2967   2968   3054   3055   3142   3143   3242   3243   3470   3471   3579   3580   3692   3693
            3804   3805   3907   3908   3909   3910   3911   4020   4021   4126   4127   4225   4226   4227
            4333   4334   4439   4440   4542   4543   4843   4844   4957   4958   4959   4960   4961   5152
            5153   5154   5155   5156   5157   5158   5159   5160   5511   5512   5622   5623   5870   5871
            5970   5971   6056   6057   6150   6151   6255   6256   6363   6364   6445   6446   6592   6593
            6736   6737   6911   6912
TMSGC       7109
```

```
                                                  TITLE   DFPTA4
                                                  SUBTTL  EBUS/MPROC 2901 Tests (Part 1)

                                                  SALL

                                                  SEARCH  DFPTA,MONSYM

                                          ; INTERN's

                                                  ENTRY   TSTA1,TSTA2,TSTA3,TSTA4,TSTA5,TSTA6,TSTA7,TSTA10
                                                  ENTRY   TSTA11,TSTA12,TSTA13,TSTA14,TSTA15,TSTA16,TSTA17,TSTA20
                                                  ENTRY   TSTA21,TSTA22,TSTA23,TSTA24,TSTA25,TSTA26,TSTA27,TSTA30

                                          ; EXTERN's

                                                  EXTERN  TSTA31,TSTA32,TSTA33,TSTA34

                                          ; EXTERN's located in DFPTA1.MAC (Mainline code..)

                                                  EXTERN  TLOAD,TRACE,TSTSUB,BEXEC,BBPNT,CEBUF,AEBUF,SSPNT,SCOSW

                                          ; EXTERN's located in DFPTAI.MAC (Port Device Handling Routines module)

                                                  EXTERN  IPACLR

                                          ; EXTERN's located in DFPTAU.MAC (Utility Routines module)

                                                  EXTERN  SCOSW


                                          ;#**********************************************************************
                                          ; Z4 - Address for use in DDT
                                          ;#**********************************************************************

 000000'                                  Z4:                                     ; address of 00000'

```

```
                                        SUBTTL  2901 Tests

                                ;#*********************************************************************
                                ; These tests do arithmetic and logical operations using the 2901's
                                ; exclusively, not using the D inputs or relying on any logic to gate
                                ; data onto the MBus and then into the 2901's.  These tests move data
                                ; around in the internal registers of the 2901's to accomplish their
                                ; testing, and data is gated into the EBUF for verification.
                                ;#*********************************************************************

                                ;#*********************************************************************
                                ;* Test 1 - 2901 0's Test
                                ;
                                ; Description:  Verify that xx AND 0's ==> 0's. The source inputs
                                ;               specified  are  any  in  the  form xx and 0.  The
                                ;               destinations are any specifying that the Y output
                                ;               is put on the MBUS which is then  gated  into the
                                ;               EBUF and  verified.  The  function  used  is  AND
                                ;               except  for  one  combination using NAND which is
                                ;               equivalent.
                                ;
                                ; Procedure:    Clear Port
                                ;               Load microcode
                                ;
                                ;               Repeat for locations 0-42:
                                ;
                                ;                 Execute a microinstruction (JMAP)
                                ;                 Read EBUF
                                ;                 Verify data is correct (0's)
                                ;
                                ; Failure:      ---
                                ;#*********************************************************************

                                ; Test data

 000000' 254 00 0 00 000011'    TSTA1:  JRST    TG1                     ; go start test
 000001' 420402  000001                 EBUS!ALU!NDMP!ZALU!1            ; test mask
 000002' 000144' 006615'                T1M,,[ASCIZ ^2901 0's Test^]
 000003' 006620' 000000                 [EXP MLAST!E23],,0
 000004' 000000  000253'                TSTA2                           ; failure test table
 000005' 000000  000636'                TSTA3                           ;   ...
 000006' 000000  000720'                TSTA4
 000007' 000000  000773'                TSTA5
 000010' 777777  777777                 -1

                                ; Start test

 000011' 201 00 0 00 000000'    TG1:    MOVEI   Z4                      ; get address of module start
 000012' 260 17 0 00 000000*            GO      TRACE                   ; handle trace output
 000013' 201 01 0 00 000144'            MOVEI   1,T1M                   ; set up microcode address
 000014' 260 17 0 00 000000*            GO      TLOAD                   ; load/verify it
 000015' 263 17 0 00 000000             RTN                             ; failed - exit test
```

```


                                          ; Initialization

000016' 400 15 0 00 000000        TL1:    SETZ    ERFLG,                  ; clear error flag
000017' 260 17 0 00 000000*               GO      IPACLR                  ; clear port
000020' 402 00 0 00 000000*               SETZM   TSTSUB                  ; initialize subtest number
000021' 201 06 0 00 000033'               MOVEI   6,TS1                   ; get sstep table address

                                          ; Loop on single step table entries

000022' 260 17 0 00 000000*       TA1:    GO      BEXEC                   ; execute table entry
000023' 254 00 0 00 000032'               JRST    TX1                     ; end of sstep table
000024' 254 00 0 00 000022'               JRST    TA1                     ; keep looping after call
000025' 474 15 0 00 000000                SETO    ERFLG,                  ; error - EBUF contents incorrect

                                          ; Handle error printouts and scope looping

000026' 027 00 0 00 000142'               SCOPER  MA1                     ; print error message
000027' 254 00 0 00 000016'               JRST    TL1                     ; loop on error
000030' 254 00 0 00 000032'               JRST    TX1                     ; altmode exit
000031' 322 15 0 00 000022'               JUMPE   ERFLG,TA1               ; do next sstep table entry

                                          ; End of test

000032' 263 17 0 00 000000        TX1:    RTN                             ; return

                                          ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

000033' 100100  000000            TS1:    ATABLE  (SSSTRT,1,0,0,0)
000034' 000000  000000
000035' 100100  010001                    ATABLE  (SSSTRT,1,1,1,0)
000036' 000000  000000
000037' 100100  020002                    ATABLE  (SSSTRT,1,2,2,0)
000040' 000000  000000
000041' 100100  030003                    ATABLE  (SSSTRT,1,3,3,0)
000042' 000000  000000
000043' 100100  040004                    ATABLE  (SSSTRT,1,4,4,0)
000044' 000000  000000
000045' 100100  050005                    ATABLE  (SSSTRT,1,5,5,0)
000046' 000000  000000
000047' 100100  060006                    ATABLE  (SSSTRT,1,6,6,0)
000050' 000000  000000
000051' 100100  070007                    ATABLE  (SSSTRT,1,7,7,0)
000052' 000000  000000
000053' 100100  100010                    ATABLE  (SSSTRT,1,10,10,0)
000054' 000000  000000
000055' 100100  110011                    ATABLE  (SSSTRT,1,11,11,0)
000056' 000000  000000
000057' 100100  120012                    ATABLE  (SSSTRT,1,12,12,0)
000060' 000000  000000
000061' 100100  130013                    ATABLE  (SSSTRT,1,13,13,0)
000062' 000000  000000
000063' 100100  140014                    ATABLE  (SSSTRT,1,14,14,0)
000064' 000000  000000
000065' 100100  150015                    ATABLE  (SSSTRT,1,15,15,0)
```

```
 144  000066' 000000  000000
 145  000067' 100100  160016                  ATABLE  (SSSTRT,1,16,16,0)
 146  000070' 000000  000000
 147  000071' 100100  170017                  ATABLE  (SSSTRT,1,17,17,0)
 148  000072' 000000  000000
 149  000073' 100100  200020                  ATABLE  (SSSTRT,1,20,20,0)
 150  000074' 000000  000000
 151  000075' 100100  210021                  ATABLE  (SSSTRT,1,21,21,0)
 152  000076' 000000  000000
 153  000077' 100100  220022                  ATABLE  (SSSTRT,1,22,22,0)
 154  000100' 000000  000000
 155  000101' 100100  230023                  ATABLE  (SSSTRT,1,23,23,0)
 156  000102' 000000  000000
 157  000103' 100100  240024                  ATABLE  (SSSTRT,1,24,24,0)
 158  000104' 000000  000000
 159  000105' 100100  250025                  ATABLE  (SSSTRT,1,25,25,0)
 160  000106' 000000  000000
 161  000107' 100100  260026                  ATABLE  (SSSTRT,1,26,26,0)
 162  000110' 000000  000000
 163  000111' 100100  270027                  ATABLE  (SSSTRT,1,27,27,0)
 164  000112' 000000  000000
 165  000113' 100100  300030                  ATABLE  (SSSTRT,1,30,30,0)
 166  000114' 000000  000000
 167  000115' 100100  310031                  ATABLE  (SSSTRT,1,31,31,0)
 168  000116' 000000  000000
 169  000117' 100100  320032                  ATABLE  (SSSTRT,1,32,32,0)
 170  000120' 000000  000000
 171  000121' 100100  330033                  ATABLE  (SSSTRT,1,33,33,0)
 172  000122' 000000  000000
 173  000123' 100100  340034                  ATABLE  (SSSTRT,1,34,34,0)
 174  000124' 000000  000000
 175  000125' 100100  350035                  ATABLE  (SSSTRT,1,35,35,0)
 176  000126' 000000  000000
 177  000127' 100100  360036                  ATABLE  (SSSTRT,1,36,36,0)
 178  000130' 000000  000000
 179  000131' 100100  370037                  ATABLE  (SSSTRT,1,37,37,0)
 180  000132' 000000  000000
 181  000133' 100100  400040                  ATABLE  (SSSTRT,1,40,40,0)
 182  000134' 000000  000000
 183  000135' 100100  410041                  ATABLE  (SSSTRT,1,41,41,0)
 184  000136' 000000  000000
 185  000137' 100100  420042                  ATABLE  (SSSTRT,1,42,42,0)
 186  000140' 000000  000000
 187  000141' 000000  000000                  ATABLE  (SSLAST)
 188
 189                                  ; Error messages
 190
 191  000142' 140000  006621'         MA1:    MSG!TXNOT![ASCIZ /Can't generate 0's out of the 2901's (result in EBUF)/]
 192  000143' 000000000000#                   LAST!CALL!TXALL!BBPNT           ; print sequencer data
 193
 194                                  ; Microcode:
 195
 196                                  ;       AND with Destination 0
 197
 198  000144' 000000  002004          T1M:    MWORD   <ADDR=0,JMAP,J=0,SDO,AND,D=0,OENA,SELE,MGC=4>   ; 0
```

```
000145' 740000  005040
000146' 000100  012004                  MWORD   <JMAP,J=1,SOQ,AND,D=0,OENA,SELE,MGC=4>          ; 1
000147' 240000  005040
000150' 000200  022004                  MWORD   <JMAP,J=2,SOB,AND,D=0,OENA,SELE,MGC=4>          ; 2
000151' 340000  005040
000152' 000300  032004                  MWORD   <JMAP,J=3,SOA,AND,D=0,OENA,SELE,MGC=4>          ; 3
000153' 440000  005040

                                ;       AND with Destination 1

000154' 000400  042004                  MWORD   <JMAP,J=4,SDO,AND,D=1,OENA,SELE,MGC=4>          ; 4
000155' 741000  005040
000156' 000500  052004                  MWORD   <JMAP,J=5,SOQ,AND,D=1,OENA,SELE,MGC=4>          ; 5
000157' 241000  005040
000160' 000600  062004                  MWORD   <JMAP,J=6,SOB,AND,D=1,OENA,SELE,MGC=4>          ; 6
000161' 341000  005040
000162' 000700  072004                  MWORD   <JMAP,J=7,SOA,AND,D=1,OENA,SELE,MGC=4>          ; 7
000163' 441000  005040

                                ;       AND with Destination 3

000164' 001000  102004                  MWORD   <JMAP,J=10,SDO,AND,D=3,OENA,SELE,MGC=4>         ; 10
000165' 743000  005040
000166' 001100  112004                  MWORD   <JMAP,J=11,SOQ,AND,D=3,OENA,SELE,MGC=4>         ; 11
000167' 243000  005040
000170' 001200  122004                  MWORD   <JMAP,J=12,SOB,AND,D=3,OENA,SELE,MGC=4>         ; 12
000171' 343000  005040
000172' 001300  132004                  MWORD   <JMAP,J=13,SOA,AND,D=3,OENA,SELE,MGC=4>         ; 13
000173' 443000  005040

                                ;       AND with Destination 4

000174' 001400  142004                  MWORD   <JMAP,J=14,SDO,AND,D=4,OENA,SELE,MGC=4>         ; 14
000175' 744000  005040
000176' 001500  152004                  MWORD   <JMAP,J=15,SOQ,AND,D=4,OENA,SELE,MGC=4>         ; 15
000177' 244000  005040
000200' 001600  162004                  MWORD   <JMAP,J=16,SOB,AND,D=4,OENA,SELE,MGC=4>         ; 16
000201' 344000  005040
000202' 001700  172004                  MWORD   <JMAP,J=17,SOA,AND,D=4,OENA,SELE,MGC=4>         ; 17
000203' 444000  005040

                                ;       AND with Destination 5

000204' 002000  202004                  MWORD   <JMAP,J=20,SDO,AND,D=5,OENA,SELE,MGC=4>         ; 20
000205' 745000  005040
000206' 002100  212004                  MWORD   <JMAP,J=21,SOQ,AND,D=5,OENA,SELE,MGC=4>         ; 21
000207' 245000  005040
000210' 002200  222004                  MWORD   <JMAP,J=22,SOB,AND,D=5,OENA,SELE,MGC=4>         ; 22
000211' 345000  005040
000212' 002300  232004                  MWORD   <JMAP,J=23,SOA,AND,D=5,OENA,SELE,MGC=4>         ; 23
000213' 445000  005040

                                ;       AND with Destination 6

000214' 002400  242004                  MWORD   <JMAP,J=24,SDO,AND,D=6,OENA,SELE,MGC=4>         ; 24
```

```
000215' 746000  005040
000216' 002500  252004          MWORD   <JMAP,J=25,SOQ,AND,D=6,OENA,SELE,MGC=4>        ; 25
000217' 246000  005040
000220' 002600  262004          MWORD   <JMAP,J=26,SOB,AND,D=6,OENA,SELE,MGC=4>        ; 26
000221' 346000  005040
000222' 002700  272004          MWORD   <JMAP,J=27,SOA,AND,D=6,OENA,SELE,MGC=4>        ; 27
000223' 446000  005040

                        ;       AND with Destination 7

000224' 003000  302004          MWORD   <JMAP,J=30,SDO,AND,D=7,OENA,SELE,MGC=4>        ; 30
000225' 747000  005040
000226' 003100  312004          MWORD   <JMAP,J=31,SOQ,AND,D=7,OENA,SELE,MGC=4>        ; 31
000227' 247000  005040
000230' 003200  322004          MWORD   <JMAP,J=32,SOB,AND,D=7,OENA,SELE,MGC=4>        ; 32
000231' 347000  005040
000232' 003300  332004          MWORD   <JMAP,J=33,SOA,AND,D=7,OENA,SELE,MGC=4>        ; 33
000233' 447000  005040

                        ;       NAND with Destination 0,1,3,4,5,6,7

000234' 003400  342004          MWORD   <JMAP,J=34,SDO,NAND,D=0,OENA,SELE,MGC=4>       ; 34
000235' 750000  005040
000236' 003500  352004          MWORD   <JMAP,J=35,SDO,NAND,D=1,OENA,SELE,MGC=4>       ; 35
000237' 751000  005040
000240' 003600  362004          MWORD   <JMAP,J=36,SDO,NAND,D=3,OENA,SELE,MGC=4>       ; 36
000241' 753000  005040
000242' 003700  372004          MWORD   <JMAP,J=37,SDO,NAND,D=4,OENA,SELE,MGC=4>       ; 37
000243' 754000  005040
000244' 004000  402004          MWORD   <JMAP,J=40,SDO,NAND,D=5,OENA,SELE,MGC=4>       ; 40
000245' 755000  005040
000246' 004100  412004          MWORD   <JMAP,J=41,SDO,NAND,D=6,OENA,SELE,MGC=4>       ; 41
000247' 756000  005040
000250' 004200  422004          MWORD   <JMAP,J=42,SDO,NAND,D=7,OENA,SELE,MGC=4>       ; 42
000251' 757000  005040
000252' 777777  777777          -1
```

```

                                   ;#*********************************************************************
                                   ;* Test 2 - 2901 1's Test
                                   ;
                                   ; Description:  Verify that 1's can be gated out  of  the  2901s.
                                   ;               The 1's are produced by executing pairs of micro-
                                   ;               instructions. The destinations are any specifying
                                   ;               that the Y output  is  put  on  the MBUS which is
                                   ;               then  gated  into  the  EBUF  and  verified.  The
                                   ;               functions used are XNOR, R-S, S-R, and OR.
                                   ;
                                   ; Procedure:    Clear Port
                                   ;               Load microcode
                                   ;
                                   ;               Repeat for locations 0-157 step 2:
                                   ;
                                   ;                 Execute 2 microinstructions (JMAP,JMAP)
                                   ;                 Read EBUF
                                   ;                 Verify data is correct (1's)
                                   ;
                                   ; Failure:      ---
                                   ;#*********************************************************************

                                   ; Test data

000253' 254 00 0 00 000264'        TSTA2:  JRST    TG2                         ; go start test
000254' 420402  000002                     EBUS!ALU!NDMP!ZALU!2                ; test mask
000255' 000331' 006634'                    T2M,,[ASCIZ ^2901 1's Test^]
000256' 006620' 000000                     [EXP MLAST!E23],,0
000257' 000000  000636'                    TSTA3                               ; failure test table
000260' 000000  000720'                    TSTA4                               ;   ...
000261' 000000  000773'                    TSTA5
000262' 000000  001046'                    TSTA6
000263' 777777  777777                     -1

                                   ; Start test

000264' 201 00 0 00 000000'        TG2:    MOVEI   Z4                          ; get address of module start
000265' 260 17 0 00 000012*                GO      TRACE                       ; handle trace output
000266' 201 01 0 00 000331'                MOVEI   1,T2M                       ; set up microcode address
000267' 260 17 0 00 000014*                GO      TLOAD                       ; load/verify it
000270' 263 17 0 00 000000                 RTN                                 ; failed - exit test

                                   ; Initialization

000271' 400 15 0 00 000000         TL2:    SETZ    ERFLG,                      ; clear error flag
000272' 260 17 0 00 000017*                GO      IPACLR                      ; clear port
000273' 402 00 0 00 000020*                SETZM   TSTSUB                      ; initialize subtest number
000274' 201 06 0 00 000306'                MOVEI   6,TS2                       ; get sstep table address
```

```

                                 ; Loop on single step table entries

000275' 260 17 0 00 000022*      TA2:    GO      BEXEC                    ; execute table entry
000276' 254 00 0 00 000305'              JRST    TX2                      ; end of sstep table
000277' 254 00 0 00 000275'              JRST    TA2                      ; keep looping after call
000300' 474 15 0 00 000000               SETO    ERFLG,                   ; error - EBUF data incorrect

                                 ; Handle error printouts and scope looping

000301' 027 00 0 00 000327'              SCOPER  MA2                      ; print error message
000302' 254 00 0 00 000271'              JRST    TL2                      ; loop on error
000303' 254 00 0 00 000305'              JRST    TX2                      ; altmode exit
000304' 322 15 0 00 000275'              JUMPE   ERFLG,TA2                ; do next sstep table entry

                                 ; End of test

000305' 263 17 0 00 000000       TX2:    RTN                              ; return

                                 ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

000306' 300000  000314'          TS2:    ATABLE  (SSCALL,TS2INI)          ; initialize test parameters
000307' 400000  000316'          TS2L:   ATABLE  (SSCHK,TS2SET)           ; set up single step parameters
000310' 100200  000000                   ATABLE  (SSSTRT,2,0,0,-1)        ; do 2 single steps/check results
000311' 777777  777777
000312' 500000  000307'                  ATABLE  (SSJRST,TS2L)            ; do next address
000313' 000000  000000                   ATABLE  (SSLAST)                 ; end of table

                                 ; Special routines

000314' 474 07 0 00 777777       TS2INI: SETO    7,-1                     ; initialize address
000315' 263 17 0 00 000000               RTN                              ; return

000316' 350 00 0 00 000007       TS2SET: AOS     7                        ; point to initial address
000317' 307 07 0 00 000141               CAIG    7,141                    ; done yet?
000320' 350 00 0 17 000000               AOS     (P)                      ; no - set up RTN+2
000321' 200 01 0 06 000000               MOVE    1,(6)                    ; get table entry
000322' 137 07 0 00 006637'              DPB     7,[POINT 12,1,23]        ; save initial address
000323' 350 00 0 00 000007               AOS     7                        ; point to final address
000324' 137 07 0 00 006640'              DPB     7,[POINT 12,1,35]        ; save final address
000325' 202 01 0 06 000000               MOVEM   1,(6)                    ; save it
000326' 263 17 0 00 000000               RTN                              ; return

                                 ; Error messages

000327' 140000  006641'          MA2:    MSG!TXNOT![ASCIZ /Can't generate 1's out of the 2901's (result in EBUF)/]
000330' 000000000000#                    LAST!CALL!TXALL!BBPNT            ; print sequencer data

                                 ; Microcode:

                                 ;       Source 0Q / XNOR / Destination 0

000331' 000000  010000           T2M:    MWORD   <ADDR=0,JMAP,J=1,S0Q,XNOR,D=2>               ; 0
000332' 272000  000040
000333' 000100  012004                   MWORD   <JMAP,J=1,SAQ,OR,D=0,OENA,SELE,MGC=4>        ; 1
```

```
000334' 030000  005040
000335' 000200  030000                  MWORD   <JMAP,J=3,SOQ,XNOR,D=3>                          ; 2
000336' 273000  000040
000337' 000300  032004                  MWORD   <JMAP,J=3,SAQ,OR,D=0,OENA,SELE,MGC=4>            ; 3
000340' 030000  005040


                                ;       Source 0Q / XNOR / Destination 1

000341' 000400  050000                  MWORD   <JMAP,J=5,SOQ,XNOR,D=2>                          ; 4
000342' 272000  000040
000343' 000500  052004                  MWORD   <JMAP,J=5,SAQ,OR,D=1,OENA,SELE,MGC=4>            ; 5
000344' 031000  005040
000345' 000600  070000                  MWORD   <JMAP,J=7,SOQ,XNOR,D=3>                          ; 6
000346' 273000  000040
000347' 000700  072004                  MWORD   <JMAP,J=7,SAQ,OR,D=1,OENA,SELE,MGC=4>            ; 7
000350' 031000  005040

                                ;       Source 0Q / XNOR / Destination 2

000351' 001000  110000                  MWORD   <JMAP,J=11,SOQ,XNOR,D=2>                         ; 10
000352' 272000  000040
000353' 001100  112004                  MWORD   <JMAP,J=11,SAQ,OR,D=3,OENA,SELE,MGC=4>           ; 11
000354' 033000  005040
000355' 001200  130000                  MWORD   <JMAP,J=13,SOQ,XNOR,D=3>                         ; 12
000356' 273000  000040
000357' 001300  132004                  MWORD   <JMAP,J=13,SAQ,OR,D=3,OENA,SELE,MGC=4>           ; 13
000360' 033000  005040

                                ;       Source 0Q / XNOR / Destination 4

000361' 001400  150000                  MWORD   <JMAP,J=15,SOQ,XNOR,D=2>                         ; 14
000362' 272000  000040
000363' 001500  152004                  MWORD   <JMAP,J=15,SAQ,OR,D=4,OENA,SELE,MGC=4>           ; 15
000364' 034000  005040
000365' 001600  170000                  MWORD   <JMAP,J=17,SOQ,XNOR,D=3>                         ; 16
000366' 273000  000040
000367' 001700  172004                  MWORD   <JMAP,J=17,SAQ,OR,D=4,OENA,SELE,MGC=4>           ; 17
000370' 034000  005040

                                ;       Source 0Q / XNOR / Destination 5

000371' 002000  210000                  MWORD   <JMAP,J=21,SOQ,XNOR,D=2>                         ; 20
000372' 272000  000040
000373' 002100  212004                  MWORD   <JMAP,J=21,SAQ,OR,D=5,OENA,SELE,MGC=4>           ; 21
000374' 035000  005040
000375' 002200  230000                  MWORD   <JMAP,J=23,SOQ,XNOR,D=3>                         ; 22
000376' 273000  000040
000377' 002300  232004                  MWORD   <JMAP,J=23,SAQ,OR,D=5,OENA,SELE,MGC=4>           ; 23
000400' 035000  005040

                                ;       Source 0Q / XNOR / Destination 6

000401' 002400  250000                  MWORD   <JMAP,J=25,SOQ,XNOR,D=2>                         ; 24
000402' 272000  000040
000403' 002500  252004                  MWORD   <JMAP,J=25,SAQ,OR,D=6,OENA,SELE,MGC=4>           ; 25
```

```
000404' 036000  005040
000405' 002600  270000                  MWORD   <JMAP,J=27,SOQ,XNOR,D=3>                        ; 26
000406' 273000  000040
000407' 002700  272004                  MWORD   <JMAP,J=27,SAQ,OR,D=6,OENA,SELE,MGC=4>          ; 27
000410' 036000  005040


                                ;       Source 0Q / XNOR / Destination 7

000411' 003000  310000                  MWORD   <JMAP,J=31,SOQ,XNOR,D=2>                        ; 30
000412' 272000  000040
000413' 003100  312004                  MWORD   <JMAP,J=31,SAQ,OR,D=7,OENA,SELE,MGC=4>          ; 31
000414' 037000  005040
000415' 003200  330000                  MWORD   <JMAP,J=33,SOQ,XNOR,D=3>                        ; 32
000416' 273000  000040
000417' 003300  332004                  MWORD   <JMAP,J=33,SAQ,OR,D=7,OENA,SELE,MGC=4>          ; 33
000420' 037000  005040


                                ;       Source 0A / XNOR / Destination 0,1,3,4,5,6,7

000421' 003400  350000                  MWORD   <JMAP,J=35,SOA,XNOR,D=0>                        ; 34
000422' 470000  000040
000423' 003500  352004                  MWORD   <JMAP,J=35,SAQ,OR,D=0,OENA,SELE,MGC=4>          ; 35
000424' 030000  005040
000425' 003600  370000                  MWORD   <JMAP,J=37,SOA,XNOR,D=0>                        ; 36
000426' 470000  000040
000427' 003700  372004                  MWORD   <JMAP,J=37,SAQ,OR,D=1,OENA,SELE,MGC=4>          ; 37
000430' 031000  005040
000431' 004000  410000                  MWORD   <JMAP,J=41,SOA,XNOR,D=0>                        ; 40
000432' 470000  000040
000433' 004100  412004                  MWORD   <JMAP,J=41,SAQ,OR,D=3,OENA,SELE,MGC=4>          ; 41
000434' 033000  005040
000435' 004200  430000                  MWORD   <JMAP,J=43,SOA,XNOR,D=0>                        ; 42
000436' 470000  000040
000437' 004300  432004                  MWORD   <JMAP,J=43,SAQ,OR,D=4,OENA,SELE,MGC=4>          ; 43
000440' 034000  005040
000441' 004400  450000                  MWORD   <JMAP,J=45,SOA,XNOR,D=0>                        ; 44
000442' 470000  000040
000443' 004500  452004                  MWORD   <JMAP,J=45,SAQ,OR,D=5,OENA,SELE,MGC=4>          ; 45
000444' 035000  005040
000445' 004600  470000                  MWORD   <JMAP,J=47,SOA,XNOR,D=0>                        ; 46
000446' 470000  000040
000447' 004700  472004                  MWORD   <JMAP,J=47,SAQ,OR,D=6,OENA,SELE,MGC=4>          ; 47
000450' 036000  005040
000451' 005000  510000                  MWORD   <JMAP,J=51,SOA,XNOR,D=0>                        ; 50
000452' 470000  000040
000453' 005100  512004                  MWORD   <JMAP,J=51,SAQ,OR,D=7,OENA,SELE,MGC=4>          ; 51
000454' 037000  005040


                                ;       Source 0Q / RMIN / Destination 0

000455' 005200  530000                  MWORD   <JMAP,J=53,SOQ,RMIN,D=2>                        ; 52
000456' 222000  000040
000457' 005300  532004                  MWORD   <JMAP,J=53,SAQ,OR,D=0,OENA,SELE,MGC=4>          ; 53
000460' 030000  005040
000461' 005400  550000                  MWORD   <JMAP,J=55,SOQ,RMIN,D=3>                        ; 54
```

```
000462' 223000  000040
000463' 005500  552004                 MWORD   <JMAP,J=55,SAQ,OR,D=0,OENA,SELE,MGC=4>        ; 55
000464' 030000  005040

                               ;       Source 0Q / RMIN / Destination 2

000465' 005600  570000                 MWORD   <JMAP,J=57,SOQ,RMIN,D=2>                      ; 56
000466' 222000  000040
000467' 005700  572004                 MWORD   <JMAP,J=57,SAQ,OR,D=1,OENA,SELE,MGC=4>        ; 57
000470' 031000  005040
000471' 006000  610000                 MWORD   <JMAP,J=61,SOQ,RMIN,D=3>                      ; 60
000472' 223000  000040
000473' 006100  612004                 MWORD   <JMAP,J=61,SAQ,OR,D=1,OENA,SELE,MGC=4>        ; 61
000474' 031000  005040

                               ;       Source 0Q / RMIN / Destination 3

000475' 006200  630000                 MWORD   <JMAP,J=63,SOQ,RMIN,D=2>                      ; 62
000476' 222000  000040
000477' 006300  632004                 MWORD   <JMAP,J=63,SAQ,OR,D=3,OENA,SELE,MGC=4>        ; 63
000500' 033000  005040
000501' 006400  650000                 MWORD   <JMAP,J=65,SOQ,RMIN,D=3>                      ; 64
000502' 223000  000040
000503' 006500  652004                 MWORD   <JMAP,J=65,SAQ,OR,D=3,OENA,SELE,MGC=4>        ; 65
000504' 033000  005040

                               ;       Source 0Q / RMIN / Destination 4

000505' 006600  670000                 MWORD   <JMAP,J=67,SOQ,RMIN,D=2>                      ; 66
000506' 222000  000040
000507' 006700  672004                 MWORD   <JMAP,J=67,SAQ,OR,D=4,OENA,SELE,MGC=4>        ; 67
000510' 034000  005040
000511' 007000  710000                 MWORD   <JMAP,J=71,SOQ,RMIN,D=3>                      ; 70
000512' 223000  000040
000513' 007100  712004                 MWORD   <JMAP,J=71,SAQ,OR,D=4,OENA,SELE,MGC=4>        ; 71
000514' 034000  005040

                               ;       Source 0Q / RMIN / Destination 5

000515' 007200  730000                 MWORD   <JMAP,J=73,SOQ,RMIN,D=2>                      ; 72
000516' 222000  000040
000517' 007300  732004                 MWORD   <JMAP,J=73,SAQ,OR,D=5,OENA,SELE,MGC=4>        ; 73
000520' 035000  005040
000521' 007400  750000                 MWORD   <JMAP,J=75,SOQ,RMIN,D=3>                      ; 74
000522' 223000  000040
000523' 007500  752004                 MWORD   <JMAP,J=75,SAQ,OR,D=5,OENA,SELE,MGC=4>        ; 75
000524' 035000  005040

                               ;       Source 0Q / RMIN / Destination 6

000525' 007600  770000                 MWORD   <JMAP,J=77,SOQ,RMIN,D=2>                      ; 76
000526' 222000  000040
000527' 007700  772004                 MWORD   <JMAP,J=77,SAQ,OR,D=6,OENA,SELE,MGC=4>        ; 77
000530' 036000  005040
000531' 010001  010000                 MWORD   <JMAP,J=101,SOQ,RMIN,D=3>                     ; 100
```

```
000532' 223000  000040
000533' 010101  012004                  MWORD   <JMAP,J=101,SAQ,OR,D=6,OENA,SELE,MGC=4>         ; 101
000534' 036000  005040

                                ;       Source 0Q / RMIN / Destination 7

000535' 010201  030000                  MWORD   <JMAP,J=103,S0Q,RMIN,D=2>                       ; 102
000536' 222000  000040
000537' 010301  032004                  MWORD   <JMAP,J=103,SAQ,OR,D=7,OENA,SELE,MGC=4>         ; 103
000540' 037000  005040
000541' 010401  050000                  MWORD   <JMAP,J=105,S0Q,RMIN,D=3>                       ; 104
000542' 223000  000040
000543' 010501  052004                  MWORD   <JMAP,J=105,SAQ,OR,D=7,OENA,SELE,MGC=4>         ; 105
000544' 037000  005040

                                ;       Source 0A / RMIN / Destination 0,1,3,4,5,6,7

000545' 010601  070000                  MWORD   <JMAP,J=107,S0A,RMIN,D=0>                       ; 106
000546' 420000  000040
000547' 010701  072004                  MWORD   <JMAP,J=107,SAQ,OR,D=0,OENA,SELE,MGC=4>         ; 107
000550' 030000  005040
000551' 011001  110000                  MWORD   <JMAP,J=111,S0A,RMIN,D=0>                       ; 110
000552' 420000  000040
000553' 011101  112004                  MWORD   <JMAP,J=111,SAQ,OR,D=1,OENA,SELE,MGC=4>         ; 111
000554' 031000  005040
000555' 011201  130000                  MWORD   <JMAP,J=113,S0A,RMIN,D=0>                       ; 112
000556' 420000  000040
000557' 011301  132004                  MWORD   <JMAP,J=113,SAQ,OR,D=3,OENA,SELE,MGC=4>         ; 113
000560' 033000  005040
000561' 011401  150000                  MWORD   <JMAP,J=115,S0A,RMIN,D=0>                       ; 114
000562' 420000  000040
000563' 011501  152004                  MWORD   <JMAP,J=115,SAQ,OR,D=4,OENA,SELE,MGC=4>         ; 115
000564' 034000  005040
000565' 011601  170000                  MWORD   <JMAP,J=117,S0A,RMIN,D=0>                       ; 116
000566' 420000  000040
000567' 011701  172004                  MWORD   <JMAP,J=117,SAQ,OR,D=5,OENA,SELE,MGC=4>         ; 117
000570' 035000  005040
000571' 012001  210000                  MWORD   <JMAP,J=121,S0A,RMIN,D=0>                       ; 120
000572' 420000  000040
000573' 012101  212004                  MWORD   <JMAP,J=121,SAQ,OR,D=6,OENA,SELE,MGC=4>         ; 121
000574' 036000  005040
000575' 012201  230000                  MWORD   <JMAP,J=123,S0A,RMIN,D=0>                       ; 122
000576' 420000  000040
000577' 012301  232004                  MWORD   <JMAP,J=123,SAQ,OR,D=7,OENA,SELE,MGC=4>         ; 123
000600' 037000  005040

                                ;       Source 0B / RMIN / Destination 0,1,3,4,5,6,7

000601' 012401  250000                  MWORD   <JMAP,J=125,S0B,RMIN,D=0>                       ; 124
000602' 320000  000040
000603' 012501  252004                  MWORD   <JMAP,J=125,SAQ,OR,D=0,OENA,SELE,MGC=4>         ; 125
000604' 030000  005040
000605' 012601  270000                  MWORD   <JMAP,J=127,S0B,RMIN,D=0>                       ; 126
000606' 320000  000040
000607' 012701  272004                  MWORD   <JMAP,J=127,SAQ,OR,D=1,OENA,SELE,MGC=4>         ; 127
```

```
000610' 031000  005040
000611' 013001  310000               MWORD   <JMAP,J=131,SOB,RMIN,D=0>                           ; 130
000612' 320000  000040
000613' 013101  312004               MWORD   <JMAP,J=131,SAQ,OR,D=3,OENA,SELE,MGC=4>             ; 131
000614' 033000  005040
000615' 013201  330000               MWORD   <JMAP,J=133,SOB,RMIN,D=0>                           ; 132
000616' 320000  000040
000617' 013301  332004               MWORD   <JMAP,J=133,SAQ,OR,D=4,OENA,SELE,MGC=4>             ; 133
000620' 034000  005040
000621' 013401  350000               MWORD   <JMAP,J=135,SOB,RMIN,D=0>                           ; 134
000622' 320000  000040
000623' 013501  352004               MWORD   <JMAP,J=135,SAQ,OR,D=5,OENA,SELE,MGC=4>             ; 135
000624' 035000  005040
000625' 013601  370000               MWORD   <JMAP,J=137,SOB,RMIN,D=0>                           ; 136
000626' 320000  000040
000627' 013701  372004               MWORD   <JMAP,J=137,SAQ,OR,D=6,OENA,SELE,MGC=4>             ; 137
000630' 036000  005040
000631' 014001  410000               MWORD   <JMAP,J=141,SOB,RMIN,D=0>                           ; 140
000632' 320000  000040
000633' 014101  412004               MWORD   <JMAP,J=141,SAQ,OR,D=7,OENA,SELE,MGC=4>             ; 141
000634' 037000  005040
000635' 777777  777777               -1

```

```
                                                SUBTTL  2901 Boolean Tests (Direct Input/0)

                                        ;#**********************************************************************
                                        ; These tests do the most basic testing of the 2901's.  Data is gated
                                        ; from the D inputs into the ALU.  An operation is then performed on the
                                        ; data and zero, and the result put on the Y outputs.  If these tests
                                        ; work there is some confidence in the data path of the direct input and
                                        ; output.  Further tests can then test the 16 RAM locations and ALU and
                                        ; destination control.  And further tests can then verify the Q register
                                        ; and data shifting.
                                        ;#**********************************************************************

                                        ;#**********************************************************************
                                        ;* Test 3 - 2901 OR Test - D OR 0
                                        ;
                                        ; Description:  Verify that 1's OR 0's ==> 1's
                                        ;               Verify that 0's OR 0's ==> 0's
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load microcode
                                        ;
                                        ;               Set RAR to 0
                                        ;               Execute JMAP,JMAP
                                        ;               Read EBUF, verify data is 777400,,001777
                                        ;                 (middle 16 bits are indeterminate)
                                        ;
                                        ;               Set RAR to 2
                                        ;               Execute JMAP,JMAP
                                        ;               Read EBUF, verify data is 000000,,000000
                                        ;                 (middle 16 bits are indeterminate)
                                        ;
                                        ; Failure:      ---
                                        ;#**********************************************************************

                                        ; Test data

000636' 254 00 0 00 000647'     TSTA3:  JRST    TG3                             ; go start test
000637' 420402  000003                  EBUS!ALU!NDMP!ZALU!3                    ; test mask
000640' 000707' 006654'                 T3M,,[ASCIZ ^2901 OR Test - D OR 0^]
000641' 006620' 006661'                 [EXP MLAST!E23],,[EXP MLAST!E24]
000642' 000000  000720'                 TSTA4                                   ; failure test table
000643' 000000  000773'                 TSTA5                                   ;   ...
000644' 000000  001046'                 TSTA6
000645' 000000  001130'                 TSTA7
000646' 777777  777777                  -1
```

```

                                       ; Start test

000647' 201 00 0 00 000000'    TG3:    MOVEI   Z4                   ; get address of module start
000650' 260 17 0 00 000265*            GO      TRACE                ; handle trace output
000651' 201 01 0 00 000707'            MOVEI   1,T3M                ; set up microcode address
000652' 260 17 0 00 000267*            GO      TLOAD                ; load/verify it
000653' 263 17 0 00 000000             RTN                          ; failed - exit test

                                       ; Initialization

000654' 400 15 0 00 000000     TL3:    SETZ    ERFLG,               ; clear error flag
000655' 260 17 0 00 000272*            GO      IPACLR               ; clear port
000656' 402 00 0 00 000273*            SETZM   TSTSUB               ; initialize subtest number
000657' 201 06 0 00 000677'            MOVEI   6,TS3                ; get sstep table address

                                       ; Loop on single step table entries

000660' 260 17 0 00 000275*    TA3:    GO      BEXEC                ; execute table entry
000661' 254 00 0 00 000676'            JRST    TX3                  ; end of sstep table
000662' 254 00 0 00 000660'            JRST    TA3                  ; keep looping after call
000663' 255 00 0 00 000000             JFCL                         ; error - EBUF data incorrect
000664' 200 01 0 00 000000*            MOVE    1,CEBUF              ; get correct data
000665' 404 01 0 00 006662'            AND     1,[777400,,1777]     ; mask out indeterminate bits
000666' 200 02 0 00 000000*            MOVE    2,AEBUF              ; get actual data
000667' 404 02 0 00 006662'            AND     2,[777400,,1777]     ; mask out indeterminate bits
000670' 312 01 0 00 000002             CAME    1,2                  ; result correct?
000671' 474 15 0 00 000000             SETO    ERFLG,               ; no - set error flag

                                       ; Handle error printouts and scope looping

000672' 027 00 0 00 000704'            SCOPER  MA3                  ; print error message
000673' 254 00 0 00 000654'            JRST    TL3                  ; loop on error
000674' 254 00 0 00 000676'            JRST    TX3                  ; altmode exit
000675' 322 15 0 00 000660'            JUMPE   ERFLG,TA3            ; do next sstep table entry

                                       ; End of test

000676' 263 17 0 00 000000     TX3:    RTN                          ; return

                                       ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

000677' 100200  000001         TS3:    ATABLE  (SSSTRT,2,0,1,-1)
000700' 777777  777777
000701' 100200  020003                 ATABLE  (SSSTRT,2,2,3,0)
000702' 000000  000000
000703' 000000  000000                 ATABLE  (SSLAST)

```

```

                                ; Error messages

000704' 140000  006663'         MA3:    MSG!TXNOT![ASCIZ /OR of Magic # and 0 failed/]
000705' 140000  006671'                 MSG!TXNOT![ASCIZ /(Result in EBUF - middle 16 bits indeterminate)/]
000706' 000000000000#                   LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                ; Microcode:

000707' 000000  011777          T3M:    MWORD   <ADDR=0,JMAP,J=1,SD0,OR,D=0,SKCN,MGC=1777>      ; 0
000710' 730000  240040
000711' 000100  012004                  MWORD   <JMAP,J=1,S0Q,OR,D=1,SELE,MGC=4,OENA>           ; 1
000712' 231000  005040
000713' 000200  030000                  MWORD   <JMAP,J=3,SD0,OR,D=0,SKCN,MGC=0>                ; 2
000714' 730000  240040
000715' 000300  032004                  MWORD   <JMAP,J=3,S0Q,OR,D=1,SELE,MGC=4,OENA>           ; 3
000716' 231000  005040
000717' 777777  777777                  -1
```

```

                                        ;#*********************************************************************
                                        ;* Test 4 - 2901 AND Test - D AND 0
                                        ;
                                        ; Description:   Verify that 1's AND 0's ==> 0's
                                        ;                Verify that 0's AND 0's ==> 0's
                                        ;
                                        ; Procedure:     Clear Port
                                        ;                Load microcode
                                        ;
                                        ;                Set RAR to 0
                                        ;                Execute JMAP,JMAP
                                        ;                Read EBUF, verify data is 000000,,000000
                                        ;
                                        ;                Set RAR to 2
                                        ;                Execute JMAP,JMAP
                                        ;                Read EBUF, verify data is 000000,,000000
                                        ;
                                        ; Failure:       ---
                                        ;#*********************************************************************

                                        ; Test data

000720' 254 00 0 00 000731'             TSTA4:  JRST    TG4                     ; go start test
000721' 420402  000004                          EBUS!ALU!NDMP!ZALU!4            ; test mask
000722' 000762' 006703'                         T4M,,[ASCIZ ^2901 AND Test - D AND 0^]
000723' 006620' 006661'                         [EXP MLAST!E23],,[EXP MLAST!E24]
000724' 000000  000773'                         TSTA5                           ; failure test table
000725' 000000  001046'                         TSTA6                           ;   ...
000726' 000000  001130'                         TSTA7
000727' 000000  001212'                         TSTA10
000730' 777777  777777                          -1

                                        ; Start test

000731' 201 00 0 00 000000'             TG4:    MOVEI   Z4                      ; get address of module start
000732' 260 17 0 00 000650*                     GO      TRACE                   ; handle trace output
000733' 201 01 0 00 000762'                     MOVEI   1,T4M                   ; set up microcode address
000734' 260 17 0 00 000652*                     GO      TLOAD                   ; load/verify it
000735' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

000736' 400 15 0 00 000000              TL4:    SETZ    ERFLG,                  ; clear error flag
000737' 260 17 0 00 000655*                     GO      IPACLR                  ; clear port
000740' 402 00 0 00 000656*                     SETZM   TSTSUB                  ; initialize subtest number
000741' 201 06 0 00 000753'                     MOVEI   6,TS4                   ; get sstep table address
```

```

                                       ; Loop on single step table entries

000742' 260 17 0 00 000660*    TA4:    GO      BEXEC                   ; execute table entry
000743' 254 00 0 00 000752'            JRST    TX4                     ; end of sstep table
000744' 254 00 0 00 000742'            JRST    TA4                     ; keep looping after call
000745' 474 15 0 00 000000             SETO    ERFLG,                  ; error - EBUF data incorrect

                                       ; Handle error printouts and scope looping

000746' 027 00 0 00 000760'            SCOPER  MA4                     ; print error message
000747' 254 00 0 00 000736'            JRST    TL4                     ; loop on error
000750' 254 00 0 00 000752'            JRST    TX4                     ; altmode exit
000751' 322 15 0 00 000742'            JUMPE   ERFLG,TA4               ; do next sstep table entry

                                       ; End of test

000752' 263 17 0 00 000000     TX4:    RTN                             ; return

                                       ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

000753' 100200  000001         TS4:    ATABLE  (SSSTRT,2,0,1,0)
000754' 000000  000000
000755' 100200  020003                 ATABLE  (SSSTRT,2,2,3,0)
000756' 000000  000000
000757' 000000  000000                 ATABLE  (SSLAST)

                                       ; Error messages

000760' 140000  006710'        MA4:    MSG!TXNOT![ASCIZ /AND of Magic # and 0 failed (result in EBUF)/]
000761' 000000000000#                  LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                       ; Microcode:

000762' 000000  011777         T4M:    MWORD   <ADDR=0,JMAP,J=1,SDO,AND,D=0,SKCN,MGC=1777>     ; 0
000763' 740000  240040
000764' 000100  012004                 MWORD   <JMAP,J=1,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 1
000765' 231000  005040
000766' 000200  030000                 MWORD   <JMAP,J=3,SDO,AND,D=0,SKCN,MGC=0>               ; 2
000767' 740000  240040
000770' 000300  032004                 MWORD   <JMAP,J=3,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 3
000771' 231000  005040
000772' 777777  777777                 -1
```

```
                                        ;#*******************************************************************
                                        ;* Test 5 - 2901 NOT_R_AND_S Test - D NOT_RS 0
                                        ;
                                        ; Description:  Verify that 1's NOT_R_AND_S 0's ==> 0's
                                        ;               Verify that 0's NOT_R_AND_S 0's ==> 0's
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load microcode
                                        ;
                                        ;               Set RAR to 0
                                        ;               Execute JMAP,JMAP
                                        ;               Read EBUF, verify data is 000000,,000000
                                        ;
                                        ;               Set RAR to 2
                                        ;               Execute JMAP,JMAP
                                        ;               Read EBUF, verify data is 000000,,000000
                                        ;
                                        ; Failure:      ---
                                        ;#*******************************************************************

                                        ; Test data

000773' 254 00 0 00 001004'             TSTA5:  JRST    TG5                             ; go start test
000774' 420402  000005                          EBUS!ALU!NDMP!ZALU!5                    ; test mask
000775' 001035' 006721'                         T5M,,[ASCIZ ^2901 NOT_R_AND_S Test - D NOT_RS 0^]
000776' 006620' 006661'                         [EXP MLAST!E23],,[EXP MLAST!E24]
000777' 000000  001046'                         TSTA6                                   ; failure test table
001000' 000000  001130'                         TSTA7                                   ;   ...
001001' 000000  001212'                         TSTA10
001002' 000000  001274'                         TSTA11
001003' 777777  777777                          -1

                                        ; Start test

001004' 201 00 0 00 000000'             TG5:    MOVEI   Z4                              ; get address of module start
001005' 260 17 0 00 000732*                     GO      TRACE                           ; handle trace output
001006' 201 01 0 00 001035'                     MOVEI   1,T5M                           ; set up microcode address
001007' 260 17 0 00 000734*                     GO      TLOAD                           ; load/verify it
001010' 263 17 0 00 000000                      RTN                                     ; failed - exit test

                                        ; Initialization

001011' 400 15 0 00 000000              TL5:    SETZ    ERFLG,                          ; clear error flag
001012' 260 17 0 00 000737*                     GO      IPACLR                          ; clear port
001013' 402 00 0 00 000740*                     SETZM   TSTSUB                          ; initialize subtest number
001014' 201 06 0 00 001026'                     MOVEI   6,TS5                           ; get sstep table address
```

```


                                  ; Loop on single step table entries

001015' 260 17 0 00 000742*       TA5:    GO      BEXEC                           ; execute table entry
001016' 254 00 0 00 001025'               JRST    TX5                             ; end of sstep table
001017' 254 00 0 00 001015'               JRST    TA5                             ; keep looping after call
001020' 474 15 0 00 000000                SETO    ERFLG,                          ; error - EBUF data incorrect

                                  ; Handle error printouts and scope looping

001021' 027 00 0 00 001033'               SCOPER  MA5                             ; print error message
001022' 254 00 0 00 001011'               JRST    TL5                             ; loop on error
001023' 254 00 0 00 001025'               JRST    TX5                             ; altmode exit
001024' 322 15 0 00 001015'               JUMPE   ERFLG,TA5                       ; do next sstep table entry

                                  ; End of test

001025' 263 17 0 00 000000        TX5:    RTN                                     ; return

                                  ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

001026' 100200  000001            TS5:    ATABLE  (SSSTRT,2,0,1,0)
001027' 000000  000000
001030' 100200  020003                    ATABLE  (SSSTRT,2,2,3,0)
001031' 000000  000000
001032' 000000  000000                    ATABLE  (SSLAST)

                                  ; Error messages

001033' 140000  006621'           MA5:    MSG!TXNOT![ASCIZ /Can't generate 0's out of the 2901's (result in EBUF)/]
001034' 000000000000#                     LAST!CALL!TXALL!BBPNT           ; print sequencer data


                                  ; Microcode:

001035' 000000  011777            T5M:    MWORD   <ADDR=0,JMAP,J=1,SDO,NAND,D=0,SKCN,MGC=1777>    ; 0
001036' 750000  240040
001037' 000100  012004                    MWORD   <JMAP,J=1,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 1
001040' 231000  005040
001041' 000200  030000                    MWORD   <JMAP,J=3,SDO,NAND,D=0,SKCN,MGC=0>              ; 2
001042' 750000  240040
001043' 000300  032004                    MWORD   <JMAP,J=3,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 3
001044' 231000  005040
001045' 777777  777777                    -1
```

```
                                        ;#*****************************************************************
                                        ;* Test 6 - 2901 XOR Test - D XOR 0
                                        ;
                                        ; Description:   Verify that 1's XOR 0's ==> 1's
                                        ;                Verify that 0's XOR 0's ==> 0's
                                        ;
                                        ; Procedure:     Clear Port
                                        ;                Load microcode
                                        ;
                                        ;                Set RAR to 0
                                        ;                Execute JMAP,JMAP
                                        ;                Read EBUF, verify data is 777400,,001777
                                        ;                  (middle 16 bits are indeterminate)
                                        ;
                                        ;                Set RAR to 2
                                        ;                Execute JMAP,JMAP
                                        ;                Read EBUF, verify data is 000000,,000000
                                        ;                  (middle 16 bits are indeterminate)
                                        ;
                                        ; Failure:       ---
                                        ;#*****************************************************************

                                        ; Test data

001046' 254 00 0 00 001057'             TSTA6:  JRST    TG6                     ; go start test
001047' 420402  000006                          EBUS!ALU!NDMP!ZALU!6            ; test mask
001050' 001117' 006730'                         T6M,,[ASCIZ ^2901 XOR Test - D XOR 0^]
001051' 006620' 006661'                         [EXP MLAST!E23],,[EXP MLAST!E24]
001052' 000000  001130'                         TSTA7                           ; failure test table
001053' 000000  001212'                         TSTA10                          ;   ...
001054' 000000  001274'                         TSTA11
001055' 000000  001356'                         TSTA12
001056' 777777  777777                          -1

                                        ; Start test

001057' 201 00 0 00 000000'             TG6:    MOVEI   Z4                      ; get address of module start
001060' 260 17 0 00 001005*                     GO      TRACE                   ; handle trace output
001061' 201 01 0 00 001117'                     MOVEI   1,T6M                   ; set up microcode address
001062' 260 17 0 00 001007*                     GO      TLOAD                   ; load/verify it
001063' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

001064' 400 15 0 00 000000              TL6:    SETZ    ERFLG,                  ; clear error flag
001065' 260 17 0 00 001012*                     GO      IPACLR                  ; clear port
001066' 402 00 0 00 001013*                     SETZM   TSTSUB                  ; initialize subtest number
001067' 201 06 0 00 001107'                     MOVEI   6,TS6                   ; get sstep table address
```

```



                                   ; Loop on single step table entries

001070' 260 17 0 00 001015*        TA6:    GO      BEXEC                   ; execute table entry
001071' 254 00 0 00 001106'                JRST    TX6                     ; end of sstep table
001072' 254 00 0 00 001070'                JRST    TA6                     ; keep looping after call
001073' 255 00 0 00 000000                 JFCL                            ; error - EBUF data incorrect
001074' 200 01 0 00 000664*                MOVE    1,CEBUF                 ; get correct data
001075' 404 01 0 00 006662'                AND     1,[777400,,1777]        ; mask out indeterminate bits
001076' 200 02 0 00 000666*                MOVE    2,AEBUF                 ; get actual data
001077' 404 02 0 00 006662'                AND     2,[777400,,1777]        ; mask out indeterminate bits
001100' 312 01 0 00 000002                 CAME    1,2                     ; result correct?
001101' 474 15 0 00 000000                 SETO    ERFLG,                  ; no - set error flag

                                   ; Handle error printouts and scope looping

001102' 027 00 0 00 001114'                SCOPER  MA6                     ; print error message
001103' 254 00 0 00 001064'                JRST    TL6                     ; loop on error
001104' 254 00 0 00 001106'                JRST    TX6                     ; altmode exit
001105' 322 15 0 00 001070'                JUMPE   ERFLG,TA6               ; do next sstep table entry

                                   ; End of test

001106' 263 17 0 00 00             TX6:    RTN                             ; return

                                   ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

001107' 100200  000001             TS6:    ATABLE  (SSSTRT,2,0,1,-1)
001110' 777777  777777
001111' 100200  020003                     ATABLE  (SSSTRT,2,2,3,0)
001112' 000000  000000
001113' 000000  000000                     ATABLE  (SSLAST)

                                   ; Error messages


001114' 140000  006735'            MA6:    MSG!TXNOT![ASCIZ /XOR does not yield oroper result/]
001115' 140000  006671'                    MSG!TXNOT![ASCIZ /(Result in EBUF - middle 16 bits indeterminate)/]
001116' 000000000000#                      LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                   ; Microcode:

001117' 000000  011777             T6M:    MWORD   <ADDR=0,JMAP,J=1,SD0,XOR,D=0,SKCN,MGC=1777>     ; 0
001120' 760000  240040
001121' 000100  012004                     MWORD   <JMAP,J=1,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 1
001122' 231000  005040
001123' 000200  030000                     MWORD   <JMAP,J=3,SD0,XOR,D=0,SKCN,MGC=0>               ; 2
001124' 760000  240040
001125' 000300  032004                     MWORD   <JMAP,J=3,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 3
001126' 231000  005040
001127' 777777  777777                     -1
```

```
                                        ;#*********************************************************************
                                        ;* Test 7 - 2901 XNOR Test - D XNOR 0
                                        ;
                                        ; Description:  Verify that 1's XNOR 0's ==> 0's
                                        ;               Verify that 0's XNOR 0's ==> 1's
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load microcode
                                        ;
                                        ;               Set RAR to 0
                                        ;               Execute JMAP,JMAP
                                        ;               Read EBUF, verify data is 000000,,000000
                                        ;                 (middle 16 bits are indeterminate)
                                        ;
                                        ;               Set RAR to 2
                                        ;               Execute JMAP,JMAP
                                        ;               Read EBUF, verify data is 777400,,001777
                                        ;                 (middle 16 bits are indeterminate)
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

 001130' 254 00 0 00 001141'            TSTA7:  JRST    TG7                     ; go start test
 001131' 420402  000007                         EBUS!ALU!NDMP!ZALU!7            ; test mask
 001132' 001201' 006744'                        T7M,,[ASCIZ ^2901 XNOR Test - D XNOR 0^]
 001133' 006620' 006661'                        [EXP MLAST!E23],,[EXP MLAST!E24]
 001134' 000000  001212'                        TSTA10                          ; failure test table
 001135' 000000  001274'                        TSTA11                          ;   ...
 001136' 000000  001356'                        TSTA12
 001137' 000000  001440'                        TSTA13
 001140' 777777  777777                         -1

                                        ; Start test

 001141' 201 00 0 00 000000'            TG7:    MOVEI   Z4                      ; get address of module start
 001142' 260 17 0 00 001060*                    GO      TRACE                   ; handle trace output
 001143' 201 01 0 00 001201'                    MOVEI   1,T7M                   ; set up microcode address
 001144' 260 17 0 00 001062*                    GO      TLOAD                   ; load/verify it
 001145' 263 17 0 00 000000                     RTN                             ; failed - exit test

                                        ; Initialization

 001146' 400 15 0 00 000000             TL7:    SETZ    ERFLG,                  ; clear error flag
 001147' 260 17 0 00 001065*                    GO      IPACLR                  ; clear port
 001150' 402 00 0 00 001066*                    SETZM   TSTSUB                  ; initialize subtest number
 001151' 201 06 0 00 001171'                    MOVEI   6,TS7                   ; get sstep table address
```

```


                                        ; Loop on single step table entries

        001152' 260 17 0 00 001070*     TA7:    GO      BEXEC                   ; execute table entry
        001153' 254 00 0 00 001170'             JRST    TX7                     ; end of sstep table
        001154' 254 00 0 00 001152'             JRST    TA7                     ; keep looping after call
        001155' 255 00 0 00 000000              JFCL                            ; error - EBUF data incorrect
        001156' 200 01 0 00 001074*             MOVE    1,CEBUF                 ; get correct data
        001157' 404 01 0 00 006662'             AND     1,[777400,,1777]        ; mask out indeterminate bits
        001160' 200 02 0 00 001076*             MOVE    2,AEBUF                 ; get actual data
        001161' 404 02 0 00 006662'             AND     2,[777400,,1777]        ; mask out indeterminate bits
        001162' 312 01 0 00 000002              CAME    1,2                     ; result correct?
        001163' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag

                                        ; Handle error printouts and scope looping

        001164' 027 00 0 00 001176'             SCOPER  MA7                     ; print error message
        001165' 254 00 0 00 001146'             JRST    TL7                     ; loop on error
        001166' 254 00 0 00 001170'             JRST    TX7                     ; altmode exit
        001167' 322 15 0 00 001152'             JUMPE   ERFLG,TA7               ; do next sstep table entry

                                        ; End of test

        001170' 263 17 0 00 000000      TX7:    RTN                             ; return

                                        ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

        001171' 100200  000001          TS7:    ATABLE  (SSSTRT,2,0,1,0)
        001172' 000000  000000
        001173' 100200  020003                  ATABLE  (SSSTRT,2,2,3,-1)
        001174' 777777  777777
        001175' 000000  000000                  ATABLE  (SSLAST)

                                        ; Error messages

        001176' 140000  006752'         MA7:    MSG!TXNOT![ASCIZ /XNOR does not yield proper result/]
        001177' 140000  006671'                 MSG!TXNOT![ASCIZ /(Result in EBUF - middle 16 bits indeterminate)/]
        001200' 000000000000#                   LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                        ; Microcode:

        001201' 000000  011777          T7M:    MWORD   <ADDR=0,JMAP,J=1,SD0,XNOR,D=0,SKCN,MGC=1777>    ; 0
        001202' 770000  240040
        001203' 000100  012004                  MWORD   <JMAP,J=1,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 1
        001204' 231000  005040
        001205' 000200  030000                  MWORD   <JMAP,J=3,SD0,XNOR,D=0,SKCN,MGC=0>              ; 2
        001206' 770000  240040
        001207' 000300  032004                  MWORD   <JMAP,J=3,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 3
        001210' 231000  005040
        001211' 777777  777777                  -1
```

```
                                        ;#*********************************************************************
                                        ;* Test 10 - 2901 R+S Test - D + 0
                                        ;
                                        ; Description:  Verify that 1's + 0's ==> 1's
                                        ;               Verify that 0's + 0's ==> 0's
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load microcode
                                        ;
                                        ;               Set RAR to 0
                                        ;               Execute JMAP,JMAP
                                        ;               Read EBUF, verify data is 777400,,001777
                                        ;                 (middle 16 bits are indeterminate)
                                        ;
                                        ;               Set RAR to 2
                                        ;               Execute JMAP,JMAP
                                        ;               Read EBUF, verify data is 000000,,000000
                                        ;                 (middle 16 bits are indeterminate)
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

001212' 254 00 0 00 001223'             TSTA10: JRST    TG10                    ; go start test
001213' 420402  000010                          EBUS!ALU!NDMP!ZALU!10           ; test mask
001214' 001263' 006761'                         T10M,,[ASCIZ ^2901 R+S Test - D + 0^]
001215' 006620' 006661'                         [EXP MLAST!E23],,[EXP MLAST!E24]
001216' 000000  001274'                         TSTA11                          ; failure test table
001217' 000000  001356'                         TSTA12                          ;   ...
001220' 000000  001440'                         TSTA13
001221' 000000  002077'                         TSTA14
001222' 777777  777777                          -1

                                        ; Start test

001223' 201 00 0 00 000000'             TG10:   MOVEI   Z4                      ; get address of module start
001224' 260 17 0 00 001142*                     GO      TRACE                   ; handle trace output
001225' 201 01 0 00 001263'                     MOVEI   1,T10M                  ; set up microcode address
001226' 260 17 0 00 001144*                     GO      TLOAD                   ; load/verify it
001227' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

001230' 400 15 0 00 000000              TL10:   SETZ    ERFLG,                  ; clear error flag
001231' 260 17 0 00 001147*                     GO      IPACLR                  ; clear port
001232' 402 00 0 00 001150*                     SETZM   TSTSUB                  ; initialize subtest number
001233' 201 06 0 00 001253'                     MOVEI   6,TS10                  ; get sstep table address
```

```

                                        ; Loop on single step table entries

001234' 260 17 0 00 001152*      TA10:   GO      BEXEC                   ; execute table entry
001235' 254 00 0 00 001252'              JRST    TX10                    ; end of sstep table
001236' 254 00 0 00 001234'              JRST    TA10                    ; keep looping after call
001237' 255 00 0 00 000000               JFCL                            ; error - EBUF data incorrect
001240' 200 01 0 00 001156*              MOVE    1,CEBUF                 ; get correct data
001241' 404 01 0 00 006662'              AND     1,[777400,,1777]        ; mask out indeterminate bits
001242' 200 02 0 00 001160*              MOVE    2,AEBUF                 ; get actual data
001243' 404 02 0 00 006662'              AND     2,[777400,,1777]        ; mask out indeterminate bits
001244' 312 01 0 00 000002               CAME    1,2                     ; result correct?
001245' 474 15 0 00 000000               SETO    ERFLG,                  ; no - set error flag

                                 ; Handle error printouts and scope looping

001246' 027 00 0 00 001260'              SCOPER  MA10                    ; print error message
001247' 254 00 0 00 001230'              JRST    TL10                    ; loop on error
001250' 254 00 0 00 001252'              JRST    TX10                    ; altmode exit
001251' 322 15 0 00 001234'              JUMPE   ERFLG,TA10              ; do next sstep table entry

                                 ; End of test

001252' 263 17 0 00 000000       TX10:   RTN                             ; return

                                 ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

001253' 100200  000001           TS10:   ATABLE  (SSSTRT,2,0,1,-1)
001254' 777777  777777
001255' 100200  020003                   ATABLE  (SSSTRT,2,2,3,0)
001256' 000000  000000
001257' 000000  000000                   ATABLE  (SSLAST)

                                 ; Error messages

001260' 140000  006766'          MA10:   MSG!TXNOT![ASCIZ /R+S does not yield proper result/]
001261' 140000  006671'                  MSG!TXNOT![ASCIZ /(Result in EBUF - middle 16 bits indeterminate)/]
001262' 000000000000#                    LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                 ; Microcode:

001263' 000000  011777           T10M:   MWORD   <ADDR=0,JMAP,J=1,SD0,PLUS,D=0,SKCN,MGC=1777>    ; 0
001264' 700000  240040
001265' 000100  012004                   MWORD   <JMAP,J=1,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 1
001266' 231000  005040
001267' 000200  030000                   MWORD   <JMAP,J=3,SD0,PLUS,D=0,SKCN,MGC=0>              ; 2
001270' 700000  240040
001271' 000300  032004                   MWORD   <JMAP,J=3,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 3
001272' 231000  005040
001273' 777777  777777                   -1
```

```

                                        ;#*****************************************************************
                                        ;* Test 11 - 2901 R-S Test - D - 0
                                        ;
                                        ; Description:  Verify that 1's - 0's ==> 1's (CRY enabled)
                                        ;               Verify that 0's - 0's ==> 0's (CRY enabled)
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load microcode
                                        ;
                                        ;               Set RAR to 0
                                        ;               Execute JMAP,JMAP
                                        ;               Read EBUF, verify data is 777400,,001777
                                        ;                 (middle 16 bits are indeterminate)
                                        ;
                                        ;               Set RAR to 2
                                        ;               Execute JMAP,JMAP
                                        ;               Read EBUF, verify data is 000000,,000000
                                        ;                 (middle 16 bits are indeterminate)
                                        ;
                                        ; Failure:      ---
                                        ;#*****************************************************************

                                        ; Test data

 001274' 254 00 0 00 001305'            TSTA11: JRST    TG11                    ; go start test
 001275' 420402  000011                         EBUS!ALU!NDMP!ZALU!11           ; test mask
 001276' 001345' 006775'                        T11M,,[ASCIZ ^2901 R-S Test - D - 0^]
 001277' 006620' 006661'                        [EXP MLAST!E23],,[EXP MLAST!E24]
 001300' 000000  001356'                        TSTA12                          ; failure test table
 001301' 000000  001440'                        TSTA13                          ;   ...
 001302' 000000  002077'                        TSTA14
 001303' 000000  002160'                        TSTA15
 001304' 777777  777777                         -1

                                        ; Start test

 001305' 201 00 0 00 000000'            TG11:   MOVEI   Z4                      ; get address of module start
 001306' 260 17 0 00 001224*                    GO      TRACE                   ; handle trace output
 001307' 201 01 0 00 001345'                    MOVEI   1,T11M                  ; set up microcode address
 001310' 260 17 0 00 001226*                    GO      TLOAD                   ; load/verify it
 001311' 263 17 0 00 000000                     RTN                             ; failed - exit test

                                        ; Initialization

 001312' 400 15 0 00 000000             TL11:   SETZ    ERFLG,                  ; clear error flag
 001313' 260 17 0 00 001231*                    GO      IPACLR                  ; clear port
 001314' 402 00 0 00 001232*                    SETZM   TSTSUB                  ; initialize subtest number
 001315' 201 06 0 00 001335'                    MOVEI   6,TS11                  ; get sstep table address
```

```

                                    ; Loop on single step table entries

001316' 260 17 0 00 001234*         TA11:   GO      BEXEC                   ; execute table entry
001317' 254 00 0 00 001334'                 JRST    TX11                    ; end of sstep table
001320' 254 00 0 00 001316'                 JRST    TA11                    ; keep looping after call
001321' 255 00 0 00 000000                  JFCL                            ; error - EBUF data incorrect
001322' 200 01 0 00 001240*                 MOVE    1,CEBUF                 ; get correct data
001323' 404 01 0 00 006662'                 AND     1,[777400,,1777]        ; mask out indeterminate bits
001324' 200 02 0 00 001242*                 MOVE    2,AEBUF                 ; get actual data
001325' 404 02 0 00 006662'                 AND     2,[777400,,1777]        ; mask out indeterminate bits
001326' 312 01 0 00 000002                  CAME    1,2                     ; result correct?
001327' 474 15 0 00 000000                  SETO    ERFLG,                  ; no - set error flag

                                    ; Handle error printouts and scope looping

001330' 027 00 0 00 001342'                 SCOPER  MA11                    ; print error message
001331' 254 00 0 00 001312'                 JRST    TL11                    ; loop on error
001332' 254 00 0 00 001334'                 JRST    TX11                    ; altmode exit
001333' 322 15 0 00 001316'                 JUMPE   ERFLG,TA11              ; do next sstep table entry

                                    ; End of test

001334' 263 17 0 00 000000          TX11:   RTN                             ; return

                                    ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

001335' 100200  000001              TS11:   ATABLE  (SSSTRT,2,0,1,-1)
001336' 777777  777777
001337' 100200  020003                      ATABLE  (SSSTRT,2,2,3,0)
001340' 000000  000000
001341' 000000  000000                      ATABLE  (SSLAST)

                                    ; Error messages

001342' 140000  007002'             MA11:   MSG!TXNOT![ASCIZ /R-S does not yield proper result/]
001343' 140000  006671'                     MSG!TXNOT![ASCIZ /(Result in EBUF - middle 16 bits indeterminate)/]
001344' 000000000000#                       LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                    ; Microcode:

001345' 000000  011777              T11M:   MWORD   <ADDR=0,JMAP,J=1,SDO,RMIN,D=0,CRY,SKCN,MGC=1777>; 0
001346' 720000  240440
001347' 000100  012004                      MWORD   <JMAP,J=1,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 1
001350' 231000  005040
001351' 000200  030000                      MWORD   <JMAP,J=3,SDO,RMIN,D=0,CRY,SKCN,MGC=0>          ; 2
001352' 720000  240440
001353' 000300  032004                      MWORD   <JMAP,J=3,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 3
001354' 231000  005040
001355' 777777  777777                      -1
```

```
                                      ;#*********************************************************************
                                      ;* Test 12 - 2901 S-R Test - 0 - D
                                      ;
                                      ; Description:  Verify that 0's - 1's ==> xxx
                                      ;               Verify that 0's - 0's ==> xxx
                                      ;
                                      ;               Where xxx is the ones complement of the D input.
                                      ;
                                      ; Procedure:    Clear Port
                                      ;               Load microcode
                                      ;
                                      ;               Set RAR to 0
                                      ;               Execute JMAP,JMAP
                                      ;               Read EBUF, verify data is 000000,,000000
                                      ;                 (middle 16 bits are indeterminate)
                                      ;
                                      ;               Set RAR to 2
                                      ;               Execute JMAP,JMAP
                                      ;               Read EBUF, verify data is 777400,,001777
                                      ;                 (middle 16 bits are indeterminate)
                                      ;
                                      ; Failure:      ---
                                      ;#*********************************************************************

                                      ; Test data

001356' 254 00 0 00 001367'           TSTA12: JRST    TG12                    ; go start test
001357' 420402  000012                        EBUS!ALU!NDMP!ZALU!12           ; test mask
001360' 001427' 007011'                       T12M,,[ASCIZ ^2901 S-R Test - 0 - D^]
001361' 006620' 006661'                       [EXP MLAST!E23],,[EXP MLAST!E24]
001362' 000000  001440'                       TSTA13                          ; failure test table
001363' 000000  002077'                       TSTA14                          ;   ...
001364' 000000  002160'                       TSTA15
001365' 000000  002374'                       TSTA16
001366' 777777  777777                        -1

                                      ; Start test

001367' 201 00 0 00 000000'           TG12:   MOVEI   Z4                      ; get address of module start
001370' 260 17 0 00 001306*                   GO      TRACE                   ; handle trace output
001371' 201 01 0 00 001427'                   MOVEI   1,T12M                  ; set up microcode address
001372' 260 17 0 00 001310*                   GO      TLOAD                   ; load/verify it
001373' 263 17 0 00 000000                    RTN                             ; failed - exit test

                                      ; Initialization

001374' 400 15 0 00 000000            TL12:   SETZ    ERFLG,                  ; clear error flag
001375' 260 17 0 00 001313*                   GO      IPACLR                  ; clear port
001376' 402 00 0 00 001314*                   SETZM   TSTSUB                  ; initialize subtest number
001377' 201 06 0 00 001417'                   MOVEI   6,TS12                  ; get sstep table address
```

```


                                ; Loop on single step table entries

001400' 260 17 0 00 001316*     TA12:   GO      BEXEC                   ; execute table entry
001401' 254 00 0 00 001416'             JRST    TX12                    ; end of sstep table
001402' 254 00 0 00 001400'             JRST    TA12                    ; keep looping after call
001403' 255 00 0 00 000000              JFCL                            ; error - EBUF data incorrect
001404' 200 01 0 00 001322*             MOVE    1,CEBUF                 ; get correct data
001405' 404 01 0 00 006662'             AND     1,[777400,,1777]        ; mask out indeterminate bits
001406' 200 02 0 00 001324*             MOVE    2,AEBUF                 ; get actual data
001407' 404 02 0 00 006662'             AND     2,[777400,,1777]        ; mask out indeterminate bits
001410' 312 01 0 00 000002              CAME    1,2                     ; result correct?
001411' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag

                                ; Handle error printouts and scope looping

001412' 027 00 0 00 001424'             SCOPER  MA12                    ; print error message
001413' 254 00 0 00 001374'             JRST    TL12                    ; loop on error
001414' 254 00 0 00 001416'             JRST    TX12                    ; altmode exit
001415' 322 15 0 00 001400'             JUMPE   ERFLG,TA12              ; do next sstep table entry

                                ; End of test

001416' 263 17 0 00 000000      TX12:   RTN                             ; return

                                ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

001417' 100200  000001          TS12:   ATABLE  (SSSTRT,2,0,1,0)
001420' 000000  000000
001421' 100200  020003                  ATABLE  (SSSTRT,2,2,3,-1)
001422' 777777  777777
001423' 000000  000000                  ATABLE  (SSLAST)

                                ; Error messages

001424' 140000  007016'         MA12:   MSG!TXNOT![ASCIZ /S-R does not yield proper result/]
001425' 140000  006671'                 MSG!TXNOT![ASCIZ /(Result in EBUF - middle 16 bits indeterminate)/]
001426' 000000000000#                   LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                ; Microcode:

001427' 000000  011777          T12M:   MWORD   <ADDR=0,JMAP,J=1,SD0,SMIN,D=0,SKCN,MGC=1777>    ; 0
001430' 710000  240040
001431' 000100  012004                  MWORD   <JMAP,J=1,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 1
001432' 231000  005040
001433' 000200  030000                  MWORD   <JMAP,J=3,SD0,SMIN,D=0,SKCN,MGC=0>              ; 2
001434' 710000  240040
001435' 000300  032004                  MWORD   <JMAP,J=3,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 3
001436' 231000  005040
001437' 777777  777777                  -1

```

```
                                                SUBTTL  2901 Register File Tests

                                        ;#*********************************************************************
                                        ;* Test 13 - 2901 Writing/Reading Registers
                                        ;
                                        ; Description:  Verify that a data  pattern  can  be  written  to
                                        ;               Register 0,  then  different  data  placed on the
                                        ;               direct inputs, then  Register 0 read correctly by
                                        ;               specifying ALU destination code 2 (A==>Y Output).
                                        ;               The ALU should not be involved during  the  read,
                                        ;               only during the write.
                                        ;
                                        ;               Repeat for Registers 1-15.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load microcode
                                        ;
                                        ;               Set RAR to 0 (R0)
                                        ;               Execute JMAP,JMAP,JMAP
                                        ;               Read EBUF, verify data is 000000,,000000
                                        ;
                                        ;               Set RAR to 3 (R1)
                                        ;               Execute JMAP,JMAP,JMAP
                                        ;               Read EBUF, verify data is 777777,,777777
                                        ;
                                        ;               Repeat for registers 1-17 at start addresses
                                        ;                 6,11 to 133,136
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

 001440' 254 00 0 00 001451'            TSTA13: JRST    TG13                    ; go start test
 001441' 420402  000013                         EBUS!ALU!NDMP!ZALU!13           ; test mask
 001442' 001576' 007025'                        T13M,,[ASCIZ ^2901 Writing/Reading Registers^]
 001443' 006620' 000000                         [EXP MLAST!E23],,0
 001444' 000000  002077'                        TSTA14                          ; failure test table
 001445' 000000  002160'                        TSTA15                          ;   ...
 001446' 000000  002374'                        TSTA16
 001447' 000000  002653'                        TSTA17
 001450' 777777  777777                         -1

                                        ; Start test

 001451' 201 00 0 00 000000'            TG13:   MOVEI   Z4                      ; get address of module start
 001452' 260 17 0 00 001370*                    GO      TRACE                   ; handle trace output
 001453' 201 01 0 00 001576'                    MOVEI   1,T13M                  ; set up microcode address
 001454' 260 17 0 00 001372*                    GO      TLOAD                   ; load/verify it
 001455' 263 17 0 00 000000                     RTN                             ; failed - exit test
```

```

                                   ; Initialization

001456' 400 15 0 00 000000         TL13:   SETZ    ERFLG,                  ; clear error flag
001457' 260 17 0 00 001375*                GO      IPACLR                  ; clear port
001460' 402 00 0 00 001376*                SETZM   TSTSUB                  ; initialize subtest number
001461' 201 06 0 00 001473'                MOVEI   6,TS13                  ; get sstep table address

                                   ; Loop on single step table entries

001462' 260 17 0 00 001400*        TA13:   GO      BEXEC                   ; execute table entry
001463' 254 00 0 00 001472'                JRST    TX13                    ; end of sstep table
001464' 254 00 0 00 001462'                JRST    TA13                    ; keep looping after call
001465' 474 15 0 00 000000                 SETO    ERFLG,                  ; error - EBUF data incorrect

                                   ; Handle error printouts and scope looping

001466' 027 00 0 00 001574'                SCOPER  MA13                    ; print error message
001467' 254 00 0 00 001456'                JRST    TL13                    ; loop on error
001470' 254 00 0 00 001472'                JRST    TX13                    ; altmode exit
001471' 322 15 0 00 001462'                JUMPE   ERFLG,TA13              ; do next sstep table entry

                                   ; End of test

001472' 263 17 0 00 000000         TX13:   RTN                             ; return

                                   ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

001473' 100300  000002             TS13:   ATABLE  (SSSTRT,3,0,2,0)        ; Register 0
001474' 000000  000000
001475' 100300  030005                     ATABLE  (SSSTRT,3,3,5,-1)
001476' 777777  777777
001477' 100300  060010                     ATABLE  (SSSTRT,3,6,10,0)       ; Register 1
001500' 000000  000000
001501' 100300  110013                     ATABLE  (SSSTRT,3,11,13,-1)
001502' 777777  777777
001503' 100300  140016                     ATABLE  (SSSTRT,3,14,16,0)      ; Register 2
001504' 000000  000000
001505' 100300  170021                     ATABLE  (SSSTRT,3,17,21,-1)
001506' 777777  777777
001507' 100300  220024                     ATABLE  (SSSTRT,3,22,24,0)      ; Register 3
001510' 000000  000000
001511' 100300  250027                     ATABLE  (SSSTRT,3,25,27,-1)
001512' 777777  777777
001513' 100300  300032                     ATABLE  (SSSTRT,3,30,32,0)      ; Register 4
001514' 000000  000000
001515' 100300  330035                     ATABLE  (SSSTRT,3,33,35,-1)
001516' 777777  777777
001517' 100300  360040                     ATABLE  (SSSTRT,3,36,40,0)      ; Register 5
001520' 000000  000000
001521' 100300  410043                     ATABLE  (SSSTRT,3,41,43,-1)
001522' 777777  777777
001523' 100300  440046                     ATABLE  (SSSTRT,3,44,46,0)      ; Register 6
001524' 000000  000000
001525' 100300  470051                     ATABLE  (SSSTRT,3,47,51,-1)
```

```
001526' 777777  777777
001527' 100300  520054          ATABLE  (SSSTRT,3,52,54,0)      ; Register 7
001530' 000000  000000
001531' 100300  550057          ATABLE  (SSSTRT,3,55,57,-1)
001532' 777777  777777
001533' 100300  600062          ATABLE  (SSSTRT,3,60,62,0)      ; Register 10
001534' 000000  000000
001535' 100300  630065          ATABLE  (SSSTRT,3,63,65,-1)
001536' 777777  777777
001537' 100300  660070          ATABLE  (SSSTRT,3,66,70,0)      ; Register 11
001540' 000000  000000
001541' 100300  710073          ATABLE  (SSSTRT,3,71,73,-1)
001542' 777777  777777
001543' 100300  740076          ATABLE  (SSSTRT,3,74,76,0)      ; Register 12
001544' 000000  000000
001545' 100300  770101          ATABLE  (SSSTRT,3,77,101,-1)
001546' 777777  777777
001547' 100301  020104          ATABLE  (SSSTRT,3,102,104,0)    ; Register 13
001550' 000000  000000
001551' 100301  050107          ATABLE  (SSSTRT,3,105,107,-1)
001552' 777777  777777
001553' 100301  100112          ATABLE  (SSSTRT,3,110,112,0)    ; Register 14
001554' 000000  000000
001555' 100301  130115          ATABLE  (SSSTRT,3,113,115,-1)
001556' 777777  777777
001557' 100301  160120          ATABLE  (SSSTRT,3,116,120,0)    ; Register 15
001560' 000000  000000
001561' 100301  210123          ATABLE  (SSSTRT,3,121,123,-1)
001562' 777777  777777
001563' 100301  240126          ATABLE  (SSSTRT,3,124,126,0)    ; Register 16
001564' 000000  000000
001565' 100301  270131          ATABLE  (SSSTRT,3,127,131,-1)
001566' 777777  777777
001567' 100301  320134          ATABLE  (SSSTRT,3,132,134,0)    ; Register 17
001570' 000000  000000
001571' 100301  350137          ATABLE  (SSSTRT,3,135,137,-1)
001572' 777777  777777
001573' 000000  000000          ATABLE  (SSLAST)

                        ; Error messages

001574' 140000  007034' MA13:   MSG!TXNOT![ASCIZ ^Could not write/read 2901 register correctly (result in EBUF)^]
001575' 000000000000#           LAST!CALL!TXALL!BBPNT           ; print sequencer data

                        ; Microcode:    Register 0

001576' 000000  010000  T13M:   MWORD   <ADDR=0,JMAP,J=1,A=0,B=0,SDO,AND,D=3>           ; 0
001577' 743000  000040
001600' 000100  000153          MWORD   <CONT,A=0,B=0,SAB,OR,D=3,SKCN,MGC=153>          ; 1
001601' 133000  240340
001602' 000200  022004          MWORD   <JMAP,J=2,A=0,SOA,OR,D=1,SELE,MGC=4,OENA>       ; 2
001603' 431000  005040

001604' 000300  040000          MWORD   <JMAP,J=4,A=0,B=0,SOA,XNOR,D=3>                 ; 3
001605' 473000  000040
```

```
 1586   001606' 000400  000153          MWORD   <CONT,A=0,B=0,SAB,OR,D=3,SKCN,MGC=153>              ; 4
 1587   001607' 133000  240340
 1588   001610' 000500  052004          MWORD   <JMAP,J=5,A=0,SOA,OR,D=1,SELE,MGC=4,OENA>           ; 5
 1589   001611' 431000  005040
 1590
 1591                           ;       Register 1
 1592
 1593   001612' 000600  070000          MWORD   <JMAP,J=7,A=1,B=1,SDO,AND,D=3>                      ; 6
 1594   001613' 743010  400040
 1595   001614' 000700  000153          MWORD   <CONT,A=1,B=1,SAB,OR,D=3,SKCN,MGC=153>              ; 7
 1596   001615' 133010  640340
 1597   001616' 001000  102004          MWORD   <JMAP,J=10,A=1,SOA,OR,D=1,SELE,MGC=4,OENA>          ; 10
 1598   001617' 431010  005040
 1599
 1600   001620' 001100  120000          MWORD   <JMAP,J=12,A=1,B=1,SOA,XNOR,D=3>                    ; 11
 1601   001621' 473010  400040
 1602   001622' 001200  000153          MWORD   <CONT,A=1,B=1,SAB,OR,D=3,SKCN,MGC=153>              ; 12
 1603   001623' 133010  640340
 1604   001624' 001300  132004          MWORD   <JMAP,J=13,A=1,SOA,OR,D=1,SELE,MGC=4,OENA>          ; 13
 1605   001625' 431010  005040
 1606
 1607                           ;       Register 2
 1608
 1609   001626' 001400  150000          MWORD   <JMAP,J=15,A=2,B=2,SDO,AND,D=3>                     ; 14
 1610   001627' 743021  000040
 1611   001630' 001500  000153          MWORD   <CONT,A=2,B=2,SAB,OR,D=3,SKCN,MGC=153>              ; 15
 1612   001631' 133021  240340
 1613   001632' 001600  162004          MWORD   <JMAP,J=16,A=2,SOA,OR,D=1,SELE,MGC=4,OENA>          ; 16
 1614   001633' 431020  005040
 1615
 1616   001634' 001700  200000          MWORD   <JMAP,J=20,A=2,B=2,SOA,XNOR,D=3>                    ; 17
 1617   001635' 473021  000040
 1618   001636' 002000  000153          MWORD   <CONT,A=2,B=2,SAB,OR,D=3,SKCN,MGC=153>              ; 20
 1619   001637' 133021  240340
 1620   001640' 002100  212004          MWORD   <JMAP,J=21,A=2,SOA,OR,D=1,SELE,MGC=4,OENA>          ; 21
 1621   001641' 431020  005040
 1622
 1623                           ;       Register 3
 1624
 1625   001642' 002200  230000          MWORD   <JMAP,J=23,A=3,B=3,SDO,AND,D=3>                     ; 22
 1626   001643' 743031  400040
 1627   001644' 002300  000153          MWORD   <CONT,A=3,B=3,SAB,OR,D=3,SKCN,MGC=153>              ; 23
 1628   001645' 133031  640340
 1629   001646' 002400  242004          MWORD   <JMAP,J=24,A=3,SOA,OR,D=1,SELE,MGC=4,OENA>          ; 24
 1630   001647' 431030  005040
 1631
 1632   001650' 002500  260000          MWORD   <JMAP,J=26,A=3,B=3,SOA,XNOR,D=3>                    ; 25
 1633   001651' 473031  400040
 1634   001652' 002600  000153          MWORD   <CONT,A=3,B=3,SAB,OR,D=3,SKCN,MGC=153>              ; 26
 1635   001653' 133031  640340
 1636   001654' 002700  272004          MWORD   <JMAP,J=27,A=3,SOA,OR,D=1,SELE,MGC=4,OENA>          ; 27
 1637   001655' 431030  005040
 1638
 1639                           ;       Register 4
 1640
```

```
001656' 003000  310000          MWORD   <JMAP,J=31,A=4,B=4,SDO,AND,D=3>                         ; 30
001657' 743042  000040
001660' 003100  000153          MWORD   <CONT,A=4,B=4,SAB,OR,D=3,SKCN,MGC=153>                  ; 31
001661' 133042  240340
001662' 003200  322004          MWORD   <JMAP,J=32,A=4,SOA,OR,D=1,SELE,MGC=4,OENA>              ; 32
001663' 431040  005040

001664' 003300  340000          MWORD   <JMAP,J=34,A=4,B=4,SOA,XNOR,D=3>                        ; 33
001665' 473042  000040
001666' 003400  000153          MWORD   <CONT,A=4,B=4,SAB,OR,D=3,SKCN,MGC=153>                  ; 34
001667' 133042  240340
001670' 003500  352004          MWORD   <JMAP,J=35,A=4,SOA,OR,D=1,SELE,MGC=4,OENA>              ; 35
001671' 431040  005040

                        ;       Register 5

001672' 003600  370000          MWORD   <JMAP,J=37,A=5,B=5,SDO,AND,D=3>                         ; 36
001673' 743052  400040
001674' 003700  000153          MWORD   <CONT,A=5,B=5,SAB,OR,D=3,SKCN,MGC=153>                  ; 37
001675' 133052  640340
001676' 004000  402004          MWORD   <JMAP,J=40,A=5,SOA,OR,D=1,SELE,MGC=4,OENA>              ; 40
001677' 431050  005040

001700' 004100  420000          MWORD   <JMAP,J=42,A=5,B=5,SOA,XNOR,D=3>                        ; 41
001701' 473052  400040
001702' 004200  000153          MWORD   <CONT,A=5,B=5,SAB,OR,D=3,SKCN,MGC=153>                  ; 42
001703' 133052  640340
001704' 004300  432004          MWORD   <JMAP,J=43,A=5,SOA,OR,D=1,SELE,MGC=4,OENA>              ; 43
001705' 431050  005040

                        ;       Register 6

001706' 004400  450000          MWORD   <JMAP,J=45,A=6,B=6,SDO,AND,D=3>                         ; 44
001707' [illegible]063  000040
001710' 004500  000153          MWORD   <CONT,A=6,B=6,SAB,OR,D=3,SKCN,MGC=153>                  ; 45
001711' 133063  240340
001712' 004600  462004          MWORD   <JMAP,J=46,A=6,SOA,OR,D=1,SELE,MGC=4,OENA>              ; 46
001713' 431060  005040

001714' 004700  500000          MWORD   <JMAP,J=50,A=6,B=6,SOA,XNOR,D=3>                        ; 47
001715' 473063  000040
001716' 005000  000153          MWORD   <CONT,A=6,B=6,SAB,OR,D=3,SKCN,MGC=153>                  ; 50
001717' 133063  240340
001720' 005100  512004          MWORD   <JMAP,J=51,A=6,SOA,OR,D=1,SELE,MGC=4,OENA>              ; 51
001721' 431060  005040

                        ;       Register 7

001722' 005200  530000          MWORD   <JMAP,J=53,A=7,B=7,SDO,AND,D=3>                         ; 52
001723' 743073  400040
001724' 005300  000153          MWORD   <CONT,A=7,B=7,SAB,OR,D=3,SKCN,MGC=153>                  ; 53
001725' 133073  640340
001726' 005400  542004          MWORD   <JMAP,J=54,A=7,SOA,OR,D=1,SELE,MGC=4,OENA>              ; 54
001727' 431070  005040
```

```
1696  001730' 005500  560000          MWORD   <JMAP,J=56,A=7,B=7,SOA,XNOR,D=3>                 ; 55
1697  001731' 473073  400040
1698  001732' 005600  000153          MWORD   <CONT,A=7,B=7,SAB,OR,D=3,SKCN,MGC=153>           ; 56
1699  001733' 133073  640340
1700  001734' 005700  572004          MWORD   <JMAP,J=57,A=7,SOA,OR,D=1,SELE,MGC=4,OENA>       ; 57
1701  001735' 431070  005040
1702
1703                          ;       Register 10
1704
1705  001736' 006000  610000          MWORD   <JMAP,J=61,A=10,B=10,SDO,AND,D=3>                ; 60
1706  001737' 743104  000040
1707  001740' 006100  000153          MWORD   <CONT,A=10,B=10,SAB,OR,D=3,SKCN,MGC=153>         ; 61
1708  001741' 133104  240340
1709  001742' 006200  622004          MWORD   <JMAP,J=62,A=10,SOA,OR,D=1,SELE,MGC=4,OENA>      ; 62
1710  001743' 431100  005040
1711
1712  001744' 006300  640000          MWORD   <JMAP,J=64,A=10,B=10,SOA,XNOR,D=3>               ; 63
1713  001745' 473104  000040
1714  001746' 006400  000153          MWORD   <CONT,A=10,B=10,SAB,OR,D=3,SKCN,MGC=153>         ; 64
1715  001747' 133104  240340
1716  001750' 006500  652004          MWORD   <JMAP,J=65,A=10,SOA,OR,D=1,SELE,MGC=4,OENA>      ; 65
1717  001751' 431100  005040
1718
1719                          ;       Register 11
1720
1721  001752' 006600  670000          MWORD   <JMAP,J=67,A=11,B=11,SDO,AND,D=3>                ; 66
1722  001753' 743114  400040
1723  001754' 006700  000153          MWORD   <CONT,A=11,B=11,SAB,OR,D=3,SKCN,MGC=153>         ; 67
1724  001755' 133114  640340
1725  001756' 007000  702004          MWORD   <JMAP,J=70,A=11,SOA,OR,D=1,SELE,MGC=4,OENA>      ; 70
1726  001757' 431110  005040
1727
1728  001760' 007100  720000          MWORD   <JMAP,J=72,A=11,B=11,SOA,XNOR,D=3>               ; 71
1729  001761' 473114  400040
1730  001762' 007200  000153          MWORD   <CONT,A=11,B=11,SAB,OR,D=3,SKCN,MGC=153>         ; 72
1731  001763' 133114  640340
1732  001764' 007300  732004          MWORD   <JMAP,J=73,A=11,SOA,OR,D=1,SELE,MGC=4,OENA>      ; 73
1733  001765' 431110  005040
1734
1735                          ;       Register 12
1736
1737  001766' 007400  750000          MWORD   <JMAP,J=75,A=12,B=12,SDO,AND,D=3>                ; 74
1738  001767' 743125  000040
1739  001770' 007500  000153          MWORD   <CONT,A=12,B=12,SAB,OR,D=3,SKCN,MGC=153>         ; 75
1740  001771' 133125  240340
1741  001772' 007600  762004          MWORD   <JMAP,J=76,A=12,SOA,OR,D=1,SELE,MGC=4,OENA>      ; 76
1742  001773' 431120  005040
1743
1744  001774' 007701  000000          MWORD   <JMAP,J=100,A=12,B=12,SOA,XNOR,D=3>              ; 77
1745  001775' 473125  000040
1746  001776' 010000  000153          MWORD   <CONT,A=12,B=12,SAB,OR,D=3,SKCN,MGC=153>         ; 100
1747  001777' 133125  240340
1748  002000' 010101  012004          MWORD   <JMAP,J=101,A=12,SOA,OR,D=1,SELE,MGC=4,OENA>     ; 101
1749  002001' 431120  005040
1750
```

```
                                   ;       Register 13

002002' 010201  030000                     MWORD   <JMAP,J=103,A=13,B=13,SDO,AND,D=3>                    ; 102
002003' 743135  400040                     MWORD   <CONT,A=13,B=13,SAB,OR,D=3,SKCN,MGC=153>              ; 103
002004' 010300  000153
002005' 133135  640340                     MWORD   <JMAP,J=104,A=13,SOA,OR,D=1,SELE,MGC=4,OENA>          ; 104
002006' 010401  042004
002007' 431130  005040

002010' 010501  060000                     MWORD   <JMAP,J=106,A=13,B=13,SOA,XNOR,D=3>                   ; 105
002011' 473135  400040                     MWORD   <CONT,A=13,B=13,SAB,OR,D=3,SKCN,MGC=153>              ; 106
002012' 010600  000153
002013' 133135  640340                     MWORD   <JMAP,J=107,A=13,SOA,OR,D=1,SELE,MGC=4,OENA>          ; 107
002014' 010701  072004
002015' 431130  005040

                                   ;       Register 14

002016' 011001  110000                     MWORD   <JMAP,J=111,A=14,B=14,SDO,AND,D=3>                    ; 110
002017' 743146  000040                     MWORD   <CONT,A=14,B=14,SAB,OR,D=3,SKCN,MGC=153>              ; 111
002020' 011100  000153
002021' 133146  240340                     MWORD   <JMAP,J=112,A=14,SOA,OR,D=1,SELE,MGC=4,OENA>          ; 112
002022' 011201  122004
002023' 431140  005040

002024' 011301  140000                     MWORD   <JMAP,J=114,A=14,B=14,SOA,XNOR,D=3>                   ; 113
002025' 473146  000040                     MWORD   <CONT,A=14,B=14,SAB,OR,D=3,SKCN,MGC=153>              ; 114
002026' 011400  000153
002027' 133146  240340                     MWORD   <JMAP,J=115,A=14,SOA,OR,D=1,SELE,MGC=4,OENA>          ; 115
002030' 011501  152004
002031' 431140  005040

                                   ;       Register 15

002032' 011601  170000                     MWORD   <JMAP,J=117,A=15,B=15,SDO,AND,D=3>                    ; 116
002033' 743156  400040                     MWORD   <CONT,A=15,B=15,SAB,OR,D=3,SKCN,MGC=153>              ; 117
002034' 011700  000153
002035' 133156  640340                     MWORD   <JMAP,J=120,A=15,SOA,OR,D=1,SELE,MGC=4,OENA>          ; 120
002036' 012001  202004
002037' 431150  005040

002040' 012101  220000                     MWORD   <JMAP,J=122,A=15,B=15,SOA,XNOR,D=3>                   ; 121
002041' 473156  400040                     MWORD   <CONT,A=15,B=15,SAB,OR,D=3,SKCN,MGC=153>              ; 122
002042' 012200  000153
002043' 133156  640340                     MWORD   <JMAP,J=123,A=15,SOA,OR,D=1,SELE,MGC=4,OENA>          ; 123
002044' 012301  232004
002045' 431150  005040

                                   ;       Register 16

002046' 012401  250000                     MWORD   <JMAP,J=125,A=16,B=16,SDO,AND,D=3>                    ; 124
002047' 743167  000040                     MWORD   <CONT,A=16,B=16,SAB,OR,D=3,SKCN,MGC=153>              ; 125
002050' 012500  000153
002051' 133167  240340                     MWORD   <JMAP,J=126,A=16,SOA,OR,D=1,SELE,MGC=4,OENA>          ; 126
002052' 012601  262004
```

```
002053' 431160  005040

002054' 012701  300000                  MWORD   <JMAP,J=130,A=16,B=16,SOA,XNOR,D=3>                     ; 127
002055' 473167  000040
002056' 013000  000153                  MWORD   <CONT,A=16,B=16,SAB,OR,D=3,SKCN,MGC=153>                ; 130
002057' 133167  240340
002060' 013101  312004                  MWORD   <JMAP,J=131,A=16,SOA,OR,D=1,SELE,MGC=4,OENA>            ; 131
002061' 431160  005040


                                ;       Register 17

002062' 013201  330000                  MWORD   <JMAP,J=133,A=17,B=17,SDO,AND,D=3>                      ; 132
002063' 743177  400040
002064' 013300  000153                  MWORD   <CONT,A=17,B=17,SAB,OR,D=3,SKCN,MGC=153>                ; 133
002065' 133177  640340
002066' 013401  342004                  MWORD   <JMAP,J=134,A=17,SOA,OR,D=1,SELE,MGC=4,OENA>            ; 134
002067' 431170  005040

002070' 013501  360000                  MWORD   <JMAP,J=136,A=17,B=17,SOA,XNOR,D=3>                     ; 135
002071' 473177  400040
002072' 013600  000153                  MWORD   <CONT,A=17,B=17,SAB,OR,D=3,SKCN,MGC=153>                ; 136
002073' 133177  640340
002074' 013701  372004                  MWORD   <JMAP,J=137,A=17,SOA,OR,D=1,SELE,MGC=4,OENA>            ; 137
002075' 431170  005040
002076' 777777  777777                  -1
```

```
                                        ;#*********************************************************************
                                        ;* Test 14 - 2901 Q-Register Test
                                        ;
                                        ; Description:  Verify that  the Q-Register can  be  loaded  with
                                        ;               various data patterns and read back properly.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load microcode
                                        ;
                                        ;               Set RAR to 0 (R0)
                                        ;               Execute JMAP,JMAP,JMAP
                                        ;               Read EBUF, verify data is 000000,,000000
                                        ;
                                        ;               Set RAR to 3 (R1)
                                        ;               Execute JMAP,JMAP,JMAP
                                        ;               Read EBUF, verify data is 777777,,777777
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

002077' 254 00 0 00 002110'             TSTA14: JRST    TG14                    ; go start test
002100' 420402  000014                          EBUS!ALU!NDMP!ZALU!14           ; test mask
002101' 002141' 007051'                         T14M,,[ASCIZ ^2901 Q-Register Test^]
002102' 006620' 000000                          [EXP MLAST!E23],,0
002103' 000000  002160'                         TSTA15                          ; failure test table
002104' 000000  002374'                         TSTA16                          ;   ...
002105' 000000  002653'                         TSTA17
002106' 000000  003130'                         TSTA20
002107' 777777  777777                          -1

                                        ; Start test

002110' 201 00 0 00 000000'             TG14:   MOVEI   Z4                      ; get address of module start
002111' 260 17 0 00 001452*                     GO      TRACE                   ; handle trace output
002112' 201 01 0 00 002141'                     MOVEI   1,T14M                  ; set up microcode address
002113' 260 17 0 00 001454*                     GO      TLOAD                   ; load/verify it
002114' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Initialization

002115' 400 15 0 00 000000              TL14:   SETZ    ERFLG,                  ; clear error flag
002116' 260 17 0 00 001457*                     GO      IPACLR                  ; clear port
002117' 402 00 0 00 001460*                     SETZM   TSTSUB                  ; initialize subtest number
002120' 201 06 0 00 002132'                     MOVEI   6,TS14                  ; get sstep table address
```

```

                                  ; Loop on single step table entries

002121' 260 17 0 00 001462*       TA14:   GO      BEXEC                   ; execute table entry
002122' 254 00 0 00 002131'               JRST    TX14                    ; end of sstep table
002123' 254 00 0 00 002121'               JRST    TA14                    ; keep looping after call
002124' 474 15 0 00 000000                SETO    ERFLG,                  ; error - EBUF data incorrect

                                  ; Handle error printouts and scope looping

002125' 027 00 0 00 002137'               SCOPER  MA14                    ; print error message
002126' 254 00 0 00 002115'               JRST    TL14                    ; loop on error
002127' 254 00 0 00 002131'               JRST    TX14                    ; altmode exit
002130' 322 15 0 00 002121'               JUMPE   ERFLG,TA14              ; do next sstep table entry

                                  ; End of test

002131' 263 17 0 00 000000        TX14:   RTN                             ; return

                                  ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

002132' 100300  000002            TS14:   ATABLE  (SSSTRT,3,0,2,0)
002133' 000000  000000
002134' 100400  030006                    ATABLE  (SSSTRT,4,3,6,-1)
002135' 777777  777777
002136' 000000  000000                    ATABLE  (SSLAST)

                                  ; Error messages

002137' 140000  007056'           MA14:   MSG!TXNOT![ASCIZ ^Could not write/read 2901 Q-Register correctly (result in EBUF)^]
002140' 000000000000#                     LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                  ; Microcode:

002141' 000000  010000            T14M:   MWORD   <ADDR=0,JMAP,J=1,SDO,AND,D=0>                   ; 0
002142' 740000  000040
002143' 000100  000153                    MWORD   <CONT,SOQ,OR,D=0,SKCN,MGC=153>                  ; 1
002144' 230000  240340
002145' 000200  022004                    MWORD   <JMAP,J=2,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 2
002146' 231000  005040

002147' 000300  040000                    MWORD   <JMAP,J=4,SOA,AND,D=2>                          ; 3
002150' 442000  000040
002151' 000400  000000                    MWORD   <CONT,SOA,XNOR,D=0>                             ; 4
002152' 470000  000340
002153' 000500  000153                    MWORD   <CONT,SOQ,OR,D=0,SKCN,MGC=153>                  ; 5
002154' 230000  240340
002155' 000600  062004                    MWORD   <JMAP,J=6,SOQ,OR,D=1,SELE,MGC=4,OENA>           ; 6
002156' 231000  005040
002157' 777777  777777                    -1
```

```

                                        ;#*************************************************************
                                        ;* Test 15 - 2901 Register Interference Test
                                        ;
                                        ; Description:  Write  address  patterns to Registers 0-17 and Q,
                                        ;               then read the registers and  verify  that correct
                                        ;               data is read.  Any error indicates  that  it  was
                                        ;               not possible to address the registers correctly.
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load microcode
                                        ;
                                        ;               Set RAR to 0
                                        ;               Execute 16 (decimal) JMAP's times (to load 2901
                                        ;                 registers with address patterns)
                                        ;
                                        ;               Set RAR to 21
                                        ;               Execute JMAP
                                        ;               Read EBUF, verify data (register address 0-20)
                                        ;                 (middle 16 bits are indeterminate)
                                        ;
                                        ;               Do last 3 steps for registers 0-17,Q
                                        ;
                                        ; Failure:      ---
                                        ;#*************************************************************

                                        ; Test data

        002160' 254 00 0 00 002171'     TSTA15: JRST    TG15                    ; go start test
        002161' 420402  000015                  EBUS!ALU!NDMP!ZALU!15           ; test mask
        002162' 002267' 007073'                 T15M,,[ASCIZ ^2901 Register Interference Test^]
        002163' 006620' 000000                  [EXP MLAST!E23],,0
        002164' 000000  002374'                 TSTA16                          ; failure test table
        002165' 000000  002653'                 TSTA17                          ;   ...
        002166' 000000  003130'                 TSTA20
        002167' 000000  003421'                 TSTA21
        002170' 777777  777777                  -1

                                        ; Start test

        002171' 201 00 0 00 000000'     TG15:   MOVEI   Z4                      ; get address of module start
        002172' 260 17 0 00 002111*             GO      TRACE                   ; handle trace output
        002173' 201 01 0 00 002267'             MOVEI   1,T15M                  ; set up microcode address
        002174' 260 17 0 00 002113*             GO      TLOAD                   ; load/verify it
        002175' 263 17 0 00 000000              RTN                             ; failed - exit test

                                        ; Initialization

        002176' 400 15 0 00 000000      TL15:   SETZ    ERFLG,                  ; clear error flag
        002177' 260 17 0 00 002116*             GO      IPACLR                  ; clear port
        002200' 402 00 0 00 002117*             SETZM   TSTSUB                  ; initialize subtest number
        002201' 201 06 0 00 002221'             MOVEI   6,TS15                  ; get sstep table address
```

```

                                    ; Loop on single step table entries

002202' 260 17 0 00 002121*         TA15:   GO      BEXEC                   ; execute table entry
002203' 254 00 0 00 002220'                 JRST    TX15                    ; end of sstep table
002204' 254 00 0 00 002202'                 JRST    TA15                    ; keep looping after call
002205' 255 00 0 00 000000                  JFCL                            ; error - EBUF data incorrect
002206' 200 01 0 00 001404*                 MOVE    1,CEBUF                 ; get correct data
002207' 404 01 0 00 006662'                 AND     1,[777400,,1777]        ; mask out indeterminate bits
002210' 200 02 0 00 001406*                 MOVE    2,AEBUF                 ; get actual data
002211' 404 02 0 00 006662'                 AND     2,[777400,,1777]        ; mask out indeterminate bits
002212' 312 01 0 00 000002                  CAME    1,2                     ; result correct?
002213' 474 15 0 00 000000                  SETO    ERFLG,                  ; no - set error flag

                                    ; Handle error printouts and scope looping

002214' 027 00 0 00 002264'                 SCOPER  MA15                    ; print error message
002215' 254 00 0 00 002176'                 JRST    TL15                    ; loop on error
002216' 254 00 0 00 002220'                 JRST    TX15                    ; altmode exit
002217' 322 15 0 00 002202'                 JUMPE   ERFLG,TA15              ; do next sstep table entry

                                    ; End of test

002220' 263 17 0 00 000000          TX15:   RTN                             ; return

                                    ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

002221' 102200  000021              TS15:   ATABLE  (SSSTRT,^D18,0,21,0B9!0)
002222' 000000  000000
002223' 100100  220022                      ATABLE  (SSSTRT,1,22,22,1B9!1)
002224' 000400  000001
002225' 100100  230023                      ATABLE  (SSSTRT,1,23,23,2B9!2)
002226' 001000  000002
002227' 100100  240024                      ATABLE  (SSSTRT,1,24,24,3B9!3)
002230' 001400  000003
002231' 100100  250025                      ATABLE  (SSSTRT,1,25,25,4B9!4)
002232' 002000  000004
002233' 100100  260026                      ATABLE  (SSSTRT,1,26,26,5B9!5)
002234' 002400  000005
002235' 100100  270027                      ATABLE  (SSSTRT,1,27,27,6B9!6)
002236' 003000  000006
002237' 100100  300030                      ATABLE  (SSSTRT,1,30,30,7B9!7)
002240' 003400  000007
002241' 100100  310031                      ATABLE  (SSSTRT,1,31,31,10B9!10)
002242' 004000  000010
002243' 100100  320032                      ATABLE  (SSSTRT,1,32,32,11B9!11)
002244' 004400  000011
002245' 100100  330033                      ATABLE  (SSSTRT,1,33,33,12B9!12)
002246' 005000  000012
002247' 100100  340034                      ATABLE  (SSSTRT,1,34,34,13B9!13)
002250' 005400  000013
002251' 100100  350035                      ATABLE  (SSSTRT,1,35,35,14B9!14)
002252' 006000  000014
002253' 100100  360036                      ATABLE  (SSSTRT,1,36,36,15B9!15)
002254' 006400  000015
```

```
002255' 100100  370037                  ATABLE  (SSSTRT,1,37,37,16B9!16)
002256' 007000  000016
002257' 100100  400040                  ATABLE  (SSSTRT,1,40,40,17B9!17)
002260' 007400  000017
002261' 100100  410041                  ATABLE  (SSSTRT,1,41,41,20B9!20)
002262' 010000  000020
002263' 000000  000000                  ATABLE  (SSLAST)

                                ; Error messages

002264' 140000  007102'         MA15:   MSG!TXNOT![ASCIZ /Register conflict seen/]
002265' 140000  006671'                 MSG!TXNOT![ASCIZ /(Result in EBUF - middle 16 bits indeterminate)/]
002266' 000000000000#                   LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                ; Microcode:    Write the registers

002267' 000000  010000          T15M:   MWORD   <ADDR=0,JMAP,J=1,A=17,B=0,SDO,OR,D=3,SKCN,MGC=0>
002270' 733170  240040
002271' 000100  000001                  MWORD   <CONT,A=16,B=1,SDO,OR,D=3,SKCN,MGC=1>           ; 1
002272' 733160  640340
002273' 000200  000002                  MWORD   <CONT,A=15,B=2,SDO,OR,D=3,SKCN,MGC=2>           ; 2
002274' 733151  240340
002275' 000300  000003                  MWORD   <CONT,A=14,B=3,SDO,OR,D=3,SKCN,MGC=3>           ; 3
002276' 733141  640340
002277' 000400  000004                  MWORD   <CONT,A=13,B=4,SDO,OR,D=3,SKCN,MGC=4>           ; 4
002300' 733132  240340
002301' 000500  000005                  MWORD   <CONT,A=12,B=5,SDO,OR,D=3,SKCN,MGC=5>           ; 5
002302' 733122  640340
002303' 000600  000006                  MWORD   <CONT,A=11,B=6,SDO,OR,D=3,SKCN,MGC=6>           ; 6
002304' 733113  240340
002305' 000700  000007                  MWORD   <CONT,A=10,B=7,SDO,OR,D=3,SKCN,MGC=7>           ; 7
002306' 733103  640340
002307' 001000  000010                  MWORD   <CONT,A=7,B=10,SDO,OR,D=3,SKCN,MGC=10>          ; 10
002310' 733074  240340
002311' 001100  000011                  MWORD   <CONT,A=6,B=11,SDO,OR,D=3,SKCN,MGC=11>          ; 11
002312' 733064  640340
002313' 001200  000012                  MWORD   <CONT,A=5,B=12,SDO,OR,D=3,SKCN,MGC=12>          ; 12
002314' 733055  240340
002315' 001300  000013                  MWORD   <CONT,A=4,B=13,SDO,OR,D=3,SKCN,MGC=13>          ; 13
002316' 733045  640340
002317' 001400  000014                  MWORD   <CONT,A=3,B=14,SDO,OR,D=3,SKCN,MGC=14>          ; 14
002320' 733036  240340
002321' 001500  000015                  MWORD   <CONT,A=2,B=15,SDO,OR,D=3,SKCN,MGC=15>          ; 15
002322' 733026  640340
002323' 001600  000016                  MWORD   <CONT,A=1,B=16,SDO,OR,D=3,SKCN,MGC=16>          ; 16
002324' 733017  240340
002325' 001700  000017                  MWORD   <CONT,A=0,B=17,SDO,OR,D=3,SKCN,MGC=17>          ; 17
002326' 733007  640340
002327' 002000  000020                  MWORD   <CONT,A=0,B=0,SDO,OR,D=0,SKCN,MGC=20>           ; 20
002330' 730000  240340

                                ;       Read them back and verify contents

002331' 002100  212004                  MWORD   <JMAP,J=21,A=0,SOA,OR,D=1,SELE,MGC=4,OENA>      ; 21
002332' 431000  005040
```

```
 2090  002333' 002200  222004          MWORD   <JMAP,J=22,A=1,SOA,OR,D=1,SELE,MGC=4,OENA>      ; 22
 2091  002334' 431010  005040
 2092  002335' 002300  232004          MWORD   <JMAP,J=23,A=2,SOA,OR,D=1,SELE,MGC=4,OENA>      ; 23
 2093  002336' 431020  005040
 2094  002337' 002400  242004          MWORD   <JMAP,J=24,A=3,SOA,OR,D=1,SELE,MGC=4,OENA>      ; 24
 2095  002340' 431030  005040
 2096  002341' 002500  252004          MWORD   <JMAP,J=25,A=4,SOA,OR,D=1,SELE,MGC=4,OENA>      ; 25
 2097  002342' 431040  005040
 2098  002343' 002600  262004          MWORD   <JMAP,J=26,A=5,SOA,OR,D=1,SELE,MGC=4,OENA>      ; 26
 2099  002344' 431050  005040
 2100  002345' 002700  272004          MWORD   <JMAP,J=27,A=6,SOA,OR,D=1,SELE,MGC=4,OENA>      ; 27
 2101  002346' 431060  005040
 2102  002347' 003000  302004          MWORD   <JMAP,J=30,A=7,SOA,OR,D=1,SELE,MGC=4,OENA>      ; 30
 2103  002350' 431070  005040
 2104  002351' 003100  312004          MWORD   <JMAP,J=31,A=10,SOA,OR,D=1,SELE,MGC=4,OENA>     ; 31
 2105  002352' 431100  005040
 2106  002353' 003200  322004          MWORD   <JMAP,J=32,A=11,SOA,OR,D=1,SELE,MGC=4,OENA>     ; 32
 2107  002354' 431110  005040
 2108  002355' 003300  332004          MWORD   <JMAP,J=33,A=12,SOA,OR,D=1,SELE,MGC=4,OENA>     ; 33
 2109  002356' 431120  005040
 2110  002357' 003400  342004          MWORD   <JMAP,J=34,A=13,SOA,OR,D=1,SELE,MGC=4,OENA>     ; 34
 2111  002360' 431130  005040
 2112  002361' 003500  352004          MWORD   <JMAP,J=35,A=14,SOA,OR,D=1,SELE,MGC=4,OENA>     ; 35
 2113  002362' 431140  005040
 2114  002363' 003600  362004          MWORD   <JMAP,J=36,A=15,SOA,OR,D=1,SELE,MGC=4,OENA>     ; 36
 2115  002364' 431150  005040
 2116  002365' 003700  372004          MWORD   <JMAP,J=37,A=16,SOA,OR,D=1,SELE,MGC=4,OENA>     ; 37
 2117  002366' 431160  005040
 2118  002367' 004000  402004          MWORD   <JMAP,J=40,A=17,SOA,OR,D=1,SELE,MGC=4,OENA>     ; 40
 2119  002370' 431170  005040
 2120  002371' 004100  412004          MWORD   <JMAP,J=41,A=0,SOQ,OR,D=1,SELE,MGC=4,OENA>      ; 41
 2121  002372' 231000  005040
 2122  002373' 777777  777777          -1
 2123
```

```
        SUBTTL  2901 ALU Function Tests

;#**********************************************************************
; This section consists of more  complete ALU testing using all
; combinations of  source  operands and doing all ALU functions
; on them.  For the logical operations  tests, the state of the
; carry in bit should have no effect - it is randomly sprinkled
; through the tests.
;#**********************************************************************

;#**********************************************************************
;* Test 16 - 2901 OR Test
;
; Description:  Verify that 0's OR 0's ==> 0's
;               Verify that 0's OR 1's ==> 1's
;               Verify that 1's OR 1's ==> 1's
;
; Procedure:    Clear Port
;               Load microcode
;
;               Set RAR to 0
;               Execute 34. JMAP's
;               Read EBUF (Contains contents of Reg 0)
;               Verify data is correct (0's)
;
;               Set RAR to 42
;               Execute JMAP
;               Read EBUF (Contains contents of Q-Reg)
;               Verify data is correct (0's)
;
;               Set RAR to 43
;               Execute 17. JMAP's
;               Read EBUF (Contains contents of Reg 0)
;               Verify data is correct (1's)
;
;               Set RAR to 64
;               Execute JMAP
;               Read EBUF (Contains contents of Q-Reg)
;               Verify data is correct (1's)
;
;               Set RAR to 65
;               Execute 12. JMAP's
;               Read EBUF (Contains contents of Reg 0)
;               Verify data is correct (1's)
;
;               Set RAR to 101
;               Execute JMAP
;               Read EBUF (Contains contents of Q-Reg)
;               Verify data is correct (1's)
;
; Failure:      ---
;#**********************************************************************
```

```

                                   ; Test data

002374' 254 00 0 00 002405'        TSTA16: JRST    TG16                    ; go start test
002375' 420402  000016                     EBUS!ALU!NDMP!ZALU!16           ; test mask
002376' 002446' 007107'                    T16M,,[ASCIZ ^2901 OR Test^]
002377' 006620' 000000                     [EXP MLAST!E23],,0
002400' 000000  002653'                    TSTA17                          ; failure test table
002401' 000000  003130'                    TSTA20                          ;   ...
002402' 000000  003421'                    TSTA21
002403' 000000  003764'                    TSTA22
002404' 777777  777777                     -1

                                   ; Start test

002405' 201 00 0 00 000000'        TG16:   MOVEI   Z4                      ; get address of module start
002406' 260 17 0 00 002172*                GO      TRACE                   ; handle trace output
002407' 201 01 0 00 002446'                MOVEI   1,T16M                  ; set up microcode address
002410' 260 17 0 00 002174*                GO      TLOAD                   ; load/verify it
002411' 263 17 0 00 000000                 RTN                             ; failed - exit test

                                   ; Initialization

002412' 400 15 0 00 000000         TL16:   SETZ    ERFLG,                  ; clear error flag
002413' 260 17 0 00 002177*                GO      IPACLR                  ; clear port
002414' 402 00 0 00 002200*                SETZM   TSTSUB                  ; initialize subtest number
002415' 201 06 0 00 002427'                MOVEI   6,TS16                  ; get sstep table address

                                   ; Loop on single step table entries

002416' 260 17 0 00 002202*        TA16:   GO      BEXEC                   ; execute table entry
002417' 254 00 0 00 002426'                JRST    TX16                    ; end of sstep table
002420' 254 00 0 00 002416'                JRST    TA16                    ; keep looping after call
002421' 474 15 0 00 000000                 SETO    ERFLG,                  ; error - EBUF data incorrect

                                   ; Handle error printouts and scope looping

002422' 027 00 0 00 002444'                SCOPER  MA16                    ; print error message
002423' 254 00 0 00 002412'                JRST    TL16                    ; loop on error
002424' 254 00 0 00 002426'                JRST    TX16                    ; altmode exit
002425' 322 15 0 00 002416'                JUMPE   ERFLG,TA16              ; do next sstep table entry

                                   ; End of test

002426' 263 17 0 00 000000         TX16:   RTN                             ; return
```

```


                                        ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

002427' 104200  000041                  TS16:   ATABLE  (SSSTRT,^D34,0,41,0)
002430' 000000  000000
002431' 100100  420042                          ATABLE  (SSSTRT,1,42,42,0)
002432' 000000  000000
002433' 102100  430063                          ATABLE  (SSSTRT,^D17,43,63,-1)
002434' 777777  777777
002435' 100100  640064                          ATABLE  (SSSTRT,1,64,64,-1)
002436' 777777  777777
002437' 101400  650100                          ATABLE  (SSSTRT,^D12,65,100,-1)
002440' 777777  777777
002441' 100101  010101                          ATABLE  (SSSTRT,1,101,101,-1)
002442' 777777  777777
002443' 000000  000000                          ATABLE  (SSLAST)

                                        ; Error messages

002444' 140000  007112'                 MA16:   MSG!TXNOT![ASCIZ /2901 not doing an OR function properly (result in EBUF)/]
002445' 000000000000#                           LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                        ; Microcode:      Verify that 0's OR 0's ==> 0's

002446' 000000  010000                  T16M:   MWORD   <ADDR=0,JMAP,J=1,SDO,AND,D=3>     ; 0
002447' 743000  000040
002450' 000100  000000                          MWORD   <CONT,SOA,OR,D=0>                 ; 1
002451' 430000  000340
002452' 000200  000000                          MWORD   <CONT,SOA,OR,D=1>                 ; 2
002453' 431000  000340
002454' 000300  000000                          MWORD   <CONT,SOA,OR,D=2>                 ; 3
002455' 432000  000340
002456' 000400  000000                          MWORD   <CONT,SOA,OR,D=3>                 ; 4
002457' 433000  000340
002460' 000500  000000                          MWORD   <CONT,SOA,OR,D=4>                 ; 5
002461' 434000  000340
002462' 000600  000000                          MWORD   <CONT,SOA,OR,D=5>                 ; 6
002463' 435000  000340
002464' 000700  000000                          MWORD   <CONT,SOA,OR,D=6>                 ; 7
002465' 436000  000340
002466' 001000  000000                          MWORD   <CONT,SOA,OR,D=7>                 ; 10
002467' 437000  000340
002470' 001100  000000                          MWORD   <CONT,SAQ,OR,D=0,CRY>             ; 11
002471' 030000  000740
002472' 001200  000000                          MWORD   <CONT,SAQ,OR,D=1,CRY>             ; 12
002473' 031000  000740
002474' 001300  000000                          MWORD   <CONT,SAQ,OR,D=2,CRY>             ; 13
002475' 032000  000740
002476' 001400  000000                          MWORD   <CONT,SAQ,OR,D=3,CRY>             ; 14
002477' 033000  000740
002500' 001500  000000                          MWORD   <CONT,SAQ,OR,D=4,CRY>             ; 15
002501' 034000  000740
002502' 001600  000000                          MWORD   <CONT,SAQ,OR,D=5,CRY>             ; 16
002503' 035000  000740
002504' 001700  000000                          MWORD   <CONT,SAQ,OR,D=6,CRY>             ; 17
```

```
002505' 036000  000740
002506' 002000  000000          MWORD   <CONT,SAQ,OR,D=7,CRY>                   ; 20
002507' 037000  000740
002510' 002100  000000          MWORD   <CONT,SOQ,OR,D=0>                       ; 21
002511' 230000  000340
002512' 002200  000000          MWORD   <CONT,SOQ,OR,D=1>                       ; 22
002513' 231000  000340
002514' 002300  000000          MWORD   <CONT,SOQ,OR,D=2>                       ; 23
002515' 232000  000340
002516' 002400  000000          MWORD   <CONT,SOQ,OR,D=3>                       ; 24
002517' 233000  000340
002520' 002500  000000          MWORD   <CONT,SOQ,OR,D=4>                       ; 25
002521' 234000  000340
002522' 002600  000000          MWORD   <CONT,SOQ,OR,D=5>                       ; 26
002523' 235000  000340
002524' 002700  000000          MWORD   <CONT,SOQ,OR,D=6,CRY>                   ; 27
002525' 236000  000740
002526' 003000  000000          MWORD   <CONT,SOQ,OR,D=7>                       ; 30
002527' 237000  000340
002530' 003100  000000          MWORD   <CONT,SOB,OR,D=0>                       ; 31
002531' 330000  000340
002532' 003200  000000          MWORD   <CONT,SOB,OR,D=1>                       ; 32
002533' 331000  000340
002534' 003300  000000          MWORD   <CONT,SOB,OR,D=2>                       ; 33
002535' 332000  000340
002536' 003400  000000          MWORD   <CONT,SOB,OR,D=3>                       ; 34
002537' 333000  000340
002540' 003500  000000          MWORD   <CONT,SOB,OR,D=4,CRY>                   ; 35
002541' 334000  000740
002542' 003600  000000          MWORD   <CONT,SOB,OR,D=5>                       ; 36
002543' 335000  000340
002544' 003700  000000          MWORD   <CONT,SOB,OR,D=6>                       ; 37
002545' 336000  000340
002546' 004000  000000          MWORD   <CONT,SOB,OR,D=7>                       ; 40
002547' 337000  000340
002550' 004100  412004          MWORD   <JMAP,J=41,SOA,OR,D=1,SELE,MGC=4,OENA>  ; 41
002551' 431000  005040
002552' 004200  422004          MWORD   <JMAP,J=42,SOQ,OR,D=1,SELE,MGC=4,OENA>  ; 42
002553' 231000  005040

                        ;       Verify that 0's OR 1's ==> 1's

002554' 004300  440000          MWORD   <JMAP,J=44,SDO,AND,D=3>                 ; 43
002555' 743000  000040
002556' 004400  000000          MWORD   <CONT,SOA,XNOR,D=0>                     ; 44
002557' 470000  000340
002560' 004500  000000          MWORD   <CONT,SOQ,OR,D=0>                       ; 45
002561' 230000  000340
002562' 004600  000000          MWORD   <CONT,SOQ,OR,D=1>                       ; 46
002563' 231000  000340
002564' 004700  000000          MWORD   <CONT,SOQ,OR,D=3>                       ; 47
002565' 233000  000340
002566' 005000  000000          MWORD   <CONT,SOQ,OR,D=5>                       ; 50
002567' 235000  000340
002570' 005100  000000          MWORD   <CONT,SOQ,OR,D=7,CRY>                   ; 51
```

```
2331  002571' 237000  000740
2332  002572' 005200  000000                MWORD   <CONT,SOQ,OR,D=2>                ; 52
2333  002573' 232000  000340
2334  002574' 005300  000000                MWORD   <CONT,SOA,OR,D=0,CRY>            ; 53
2335  002575' 430000  000740
2336  002576' 005400  000000                MWORD   <CONT,SOA,OR,D=1,CRY>            ; 54
2337  002577' 431000  000740
2338  002600' 005500  000000                MWORD   <CONT,SOA,OR,D=2>                ; 55
2339  002601' 432000  000340
2340  002602' 005600  000000                MWORD   <CONT,SOA,OR,D=3,CRY>            ; 56
2341  002603' 433000  000740
2342  002604' 005700  000000                MWORD   <CONT,SOB,OR,D=0,CRY>            ; 57
2343  002605' 330000  000740
2344  002606' 006000  000000                MWORD   <CONT,SOB,OR,D=1>                ; 60
2345  002607' 331000  000340
2346  002610' 006100  000000                MWORD   <CONT,SOB,OR,D=2>                ; 61
2347  002611' 332000  000340
2348  002612' 006200  000000                MWORD   <CONT,SOB,OR,D=3>                ; 62
2349  002613' 333000  000340
2350  002614' 006300  632004                MWORD   <JMAP,J=63,SOA,OR,D=1,SELE,MGC=4,OENA> ; 63
2351  002615' 431000  005040
2352  002616' 006400  642004                MWORD   <JMAP,J=64,SOQ,OR,D=1,SELE,MGC=4,OENA> ; 64
2353  002617' 231000  005040
2354
2355                                ;       Verify that 1's OR 1's ==> 1's
2356
2357  002620' 006500  660000                MWORD   <JMAP,J=66,SD0,AND,D=3>          ; 65
2358  002621' 743000  000040
2359  002622' 006600  000000                MWORD   <CONT,SOA,XNOR,D=0>              ; 66
2360  002623' 470000  000340
2361  002624' 006700  000000                MWORD   <CONT,SOQ,OR,D=2>                ; 67
2362  002625' 232000  000340
2363  002626' 007000  000000                MWORD   <CONT,SAQ,OR,D=0,CRY>            ; 70
2364  002627' 030000  000740
2365  002630' 007100  000000                MWORD   <CONT,SAQ,OR,D=1>                ; 71
2366  002631' 031000  000340
2367  002632' 007200  000000                MWORD   <CONT,SAQ,OR,D=2,CRY>            ; 72
2368  002633' 032000  000740
2369  002634' 007300  000000                MWORD   <CONT,SAQ,OR,D=3>                ; 73
2370  002635' 033000  000340
2371  002636' 007400  000000                MWORD   <CONT,SAB,OR,D=0>                ; 74
2372  002637' 130000  000340
2373  002640' 007500  000000                MWORD   <CONT,SAB,OR,D=1,CRY>            ; 75
2374  002641' 131000  000740
2375  002642' 007600  000000                MWORD   <CONT,SAB,OR,D=2>                ; 76
2376  002643' 132000  000340
2377  002644' 007700  000000                MWORD   <CONT,SAB,OR,D=3>                ; 77
2378  002645' 133000  000340
2379  002646' 010001  002004                MWORD   <JMAP,J=100,SOA,OR,D=1,SELE,MGC=4,OENA> ; 100
2380  002647' 431000  005040
2381  002650' 010101  012004                MWORD   <JMAP,J=101,SOQ,OR,D=1,SELE,MGC=4,OENA> ; 101
2382  002651' 231000  005040
2383  002652' 777777  777777                -1
```

```
                                        ;#*********************************************************************
                                        ;* Test 17 - 2901 AND Test
                                        ;
                                        ; Description:  Verify that 0's AND 0's ==> 0's
                                        ;               Verify that 0's AND 1's ==> 0's
                                        ;               Verify that 1's AND 1's ==> 1's
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load microcode
                                        ;
                                        ;               Set RAR to 0
                                        ;               Execute 36. JMAP's
                                        ;               Read EBUF (Contains contents of Reg 0)
                                        ;               Verify data is correct (0's)
                                        ;
                                        ;               Set RAR to 44
                                        ;               Execute 20. JMAP's
                                        ;               Read EBUF (Contains contents of Reg 0)
                                        ;               Verify data is correct (0's)
                                        ;
                                        ;               Set RAR to 70
                                        ;               Execute 12. JMAP's
                                        ;               Read EBUF (Contains contents of Reg 0)
                                        ;               Verify data is correct (1's)
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

002653' 254 00 0 00 002664'             TSTA17: JRST    TG17                            ; go start test
002654' 420402  000017                          EBUS!ALU!NDMP!ZALU!17                   ; test mask
002655' 002717' 007126'                         T17M,,[ASCIZ ^2901 AND Test^]
002656' 006620' 000000                          [EXP MLAST!E23],,0
002657' 000000  003130'                         TSTA20                                  ; failure test table
002660' 000000  003421'                         TSTA21                                  ;   ...
002661' 000000  003764'                         TSTA22
002662' 000000  004445'                         TSTA23
002663' 777777  777777                          -1

                                        ; Start test

002664' 201 00 0 00 000000'             TG17:   MOVEI   Z4                              ; get address of module start
002665' 260 17 0 00 002406*                     GO      TRACE                           ; handle trace output
002666' 201 01 0 00 002717'                     MOVEI   1,T17M                          ; set up microcode address
002667' 260 17 0 00 002410*                     GO      TLOAD                           ; load/verify it
002670' 263 17 0 00 000000                      RTN                                     ; failed - exit test
```

```


                                          ; Initialization

 002671' 400 15 0 00 000000      TL17:   SETZ    ERFLG,                  ; clear error flag
 002672' 260 17 0 00 002413*             GO      IPACLR                  ; clear port
 002673' 402 00 0 00 002414*             SETZM   TSTSUB                  ; initialize subtest number
 002674' 201 06 0 00 002706'             MOVEI   6,TS17                  ; get sstep table address

                                 ; Loop on single step table entries

 002675' 260 17 0 00 002416*     TA17:   GO      BEXEC                   ; execute table entry
 002676' 254 00 0 00 002705'             JRST    TX17                    ; end of sstep table
 002677' 254 00 0 00 002675'             JRST    TA17                    ; keep looping after call
 002700' 474 15 0 00 000000              SETO    ERFLG,                  ; error - EBUF data incorrect

                                 ; Handle error printouts and scope looping

 002701' 027 00 0 00 002715'             SCOPER  MA17                    ; print error message
 002702' 254 00 0 00 002671'             JRST    TL17                    ; loop on error
 002703' 254 00 0 00 002705'             JRST    TX17                    ; altmode exit
 002704' 322 15 0 00 002675'             JUMPE   ERFLG,TA17              ; do next sstep table entry

                                 ; End of test

 002705' 263 17 0 00 000000      TX17:   RTN                             ; return

                                 ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

 002706' 104400  000043          TS17:   ATABLE  (SSSTRT,^D36,0,43,0)
 002707' 000000  000000
 002710' 102400  440067                  ATABLE  (SSSTRT,^D20,44,67,0)
 002711' 000000  000000
 002712' 101400  700103                  ATABLE  (SSSTRT,^D12,70,103,-1)
 002713' 777777  777777
 002714' 000000  000000                  ATABLE  (SSLAST)

                                 ; Error messages

 002715' 140000  007131'         MA17:   MSG!TXNOT![ASCIZ /2901 not doing an AND function properly (result in EBUF)/]
 002716' 000000000000#                   LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                 ; Microcode:    Verify that 0's AND 0's ==> 0's

 002717' 000000  010000          T17M:   MWORD   <ADDR=0,JMAP,J=1,B=1,SD0,AND,D=2>
 002720' 742000  400040
 002721' 000100  000000                  MWORD   <CONT,SOA,AND,D=0>              ; 1
 002722' 440000  000340
 002723' 000200  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>           ; 2
 002724' 030010  000340
 002725' 000300  000000                  MWORD   <CONT,B=1,SOA,AND,D=2>          ; 3
 002726' 442000  400340
 002727' 000400  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>           ; 4
 002730' 030010  000340
 002731' 000500  000000                  MWORD   <CONT,B=1,SOA,AND,D=3>          ; 5
 002732' 443000  400340
```

```
002733' 000600  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>                   ; 6
002734' 030010  000340
002735' 000700  000000                  MWORD   <CONT,B=1,SOA,AND,D=5,CRY>              ; 7
002736' 445000  400740
002737' 001000  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>                   ; 10
002740' 030010  000340
002741' 001100  000000                  MWORD   <CONT,B=1,SOA,AND,D=7,CRY>              ; 11
002742' 447000  400740
002743' 001200  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>                   ; 12
002744' 030010  000340
002745' 001300  000000                  MWORD   <CONT,B=1,SAQ,AND,D=2,CRY>              ; 13
002746' 042000  400740
002747' 001400  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>                   ; 14
002750' 030010  000340
002751' 001500  000000                  MWORD   <CONT,B=1,SAQ,AND,D=3,CRY>              ; 15
002752' 043000  400740
002753' 001600  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>                   ; 16
002754' 030010  000340
002755' 001700  000000                  MWORD   <CONT,B=1,SAQ,AND,D=5>                  ; 17
002756' 045000  400340
002757' 002000  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>                   ; 20
002760' 030010  000340
002761' 002100  000000                  MWORD   <CONT,B=1,SAQ,AND,D=7>                  ; 21
002762' 047000  400340
002763' 002200  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>                   ; 22
002764' 030010  000340
002765' 002300  000000                  MWORD   <CONT,B=1,SOQ,AND,D=2>                  ; 23
002766' 242000  400340
002767' 002400  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>                   ; 24
002770' 030010  000340
002771' 002500  000000                  MWORD   <CONT,B=1,SOQ,AND,D=3,CRY>              ; 25
002772' 243000  400740
002773' 002600  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>                   ; 26
002774' 030010  000340
002775' 002700  000000                  MWORD   <CONT,B=1,SOQ,AND,D=5,CRY>              ; 27
002776' 245000  400740
002777' 003000  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>                   ; 30
003000' 030010  000340
003001' 003100  000000                  MWORD   <CONT,B=1,SOQ,AND,D=7>                  ; 31
003002' 247000  400340
003003' 003200  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>                   ; 32
003004' 030010  000340
003005' 003300  000000                  MWORD   <CONT,SOB,AND,D=2>                      ; 33
003006' 342000  000340
003007' 003400  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>                   ; 34
003010' 030010  000340
003011' 003500  000000                  MWORD   <CONT,SOB,AND,D=3,CRY>                  ; 35
003012' 343000  000740
003013' 003600  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>                   ; 36
003014' 030010  000340
003015' 003700  000000                  MWORD   <CONT,SOB,AND,D=5>                      ; 37
003016' 345000  000340
003017' 004000  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>                   ; 40
003020' 030010  000340
003021' 004100  000000                  MWORD   <CONT,SOB,AND,D=7,CRY>                  ; 41
```

```
003022' 347000  000740
003023' 004200  000000                  MWORD   <CONT,A=1,SAQ,OR,D=0>              ; 42
003024' 030010  000340
003025' 004300  432004                  MWORD   <JMAP,J=43,SOQ,OR,D=1,SELE,MGC=4,OENA> ; 43
003026' 231000  005040

                                ;       Verify that 0's AND 1's ==> 0's

003027' 004400  450000                  MWORD   <JMAP,J=45,SDO,AND,B=17,D=2>       ; 44
003030' 742007  400040
003031' 004500  000000                  MWORD   <CONT,SOA,A=17,XNOR,D=0>           ; 45
003032' 470170  000340
003033' 004600  000000                  MWORD   <CONT,SOA,A=17,XNOR,D=2>           ; 46
003034' 472170  000340
003035' 004700  000000                  MWORD   <CONT,B=1,SOQ,AND,D=2,CRY>         ; 47
003036' 242000  400740
003037' 005000  000000                  MWORD   <CONT,A=1,B=17,SAB,OR,D=2>         ; 50
003040' 132017  400340
003041' 005100  000000                  MWORD   <CONT,B=1,SOQ,AND,D=3,CRY>         ; 51
003042' 243000  400740
003043' 005200  000000                  MWORD   <CONT,A=1,B=17,SAB,OR,D=2>         ; 52
003044' 132017  400340
003045' 005300  000000                  MWORD   <CONT,B=1,SOQ,AND,D=5,CRY>         ; 53
003046' 245000  400740
003047' 005400  000000                  MWORD   <CONT,A=1,B=17,SAB,OR,D=2>         ; 54
003050' 132017  400340
003051' 005500  000000                  MWORD   <CONT,B=1,SOQ,AND,D=7,CRY>         ; 55
003052' 247000  400740
003053' 005600  000000                  MWORD   <CONT,A=1,B=17,SAB,OR,D=2>         ; 56
003054' 132017  400340
003055' 005700  000000                  MWORD   <CONT,B=1,SOA,AND,D=2>             ; 57
003056' 442000  400340
003057' 006000  000000                  MWORD   <CONT,A=1,B=17,SAB,OR,D=2>         ; 60
003060' 132017  400340
003061' 006100  000000                  MWORD   <CONT,B=1,SOA,AND,D=3>             ; 61
003062' 443000  400340
003063' 006200  000000                  MWORD   <CONT,A=1,B=17,SAB,OR,D=2>         ; 62
003064' 132017  400340
003065' 006300  000000                  MWORD   <CONT,B=1,SOA,AND,D=5,CRY>         ; 63
003066' 445000  400740
003067' 006400  000000                  MWORD   <CONT,A=1,B=17,SAB,OR,D=2>         ; 64
003070' 132017  400340
003071' 006500  000000                  MWORD   <CONT,B=1,SOA,AND,D=7,CRY>         ; 65
003072' 447000  400740
003073' 006600  000000                  MWORD   <CONT,A=1,B=17,SAB,OR,D=2>         ; 66
003074' 132017  400340
003075' 006700  672004                  MWORD   <JMAP,J=67,A=17,SOA,OR,D=1,SELE,MGC=4,OENA> ; 67
003076' 431170  005040

                                ;       Verify that 1's AND 1's ==> 1's

003077' 007000  710000                  MWORD   <JMAP,J=71,SDO,AND,D=3>            ; 70
003100' 743000  000040
003101' 007100  000000                  MWORD   <CONT,SOA,XNOR,D=0>                ; 71
003102' 470000  000340
```

```
2597  003103' 007200  000000            MWORD   <CONT,SOA,XNOR,D=2>             ; 72
2598  003104' 472000  000340
2599  003105' 007300  000000            MWORD   <CONT,SAQ,AND,D=0>              ; 73
2600  003106' 040000  000340
2601  003107' 007400  000000            MWORD   <CONT,SAQ,AND,D=1,CRY>          ; 74
2602  003110' 041000  000740
2603  003111' 007500  000000            MWORD   <CONT,SAQ,AND,D=2,CRY>          ; 75
2604  003112' 042000  000740
2605  003113' 007600  000000            MWORD   <CONT,SAQ,AND,D=3,CRY>          ; 76
2606  003114' 043000  000740
2607  003115' 007700  000000            MWORD   <CONT,SAB,AND,D=0,CRY>          ; 77
2608  003116' 140000  000740
2609  003117' 010000  000000            MWORD   <CONT,SAB,AND,D=1>              ; 100
2610  003120' 141000  000340
2611  003121' 010100  000000            MWORD   <CONT,SAB,AND,D=2>              ; 101
2612  003122' 142000  000340
2613  003123' 010200  000000            MWORD   <CONT,SAB,AND,D=3>              ; 102
2614  003124' 143000  000340
2615  003125' 010301  032004            MWORD   <JMAP,J=103,SAQ,AND,D=1,SELE,MGC=4,OENA> ; 103
2616  003126' 041000  005040
2617  003127' 777777  777777            -1
```

```

                                        ;#******************************************************************
                                        ;* Test 20 - 2901 NOT_R_AND_S Test
                                        ;
                                        ; Description:  Verify that 0's NOT_R_AND_S 0's ==> 0's
                                        ;               Verify that 1's NOT_R_AND_S 1's ==> 0's
                                        ;               Verify that 0's NOT_R_AND_S 1's ==> 1's
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load microcode
                                        ;
                                        ;               Set RAR to 0
                                        ;               Execute 36. JMAP's
                                        ;               Read EBUF (Contains contents of Reg 17)
                                        ;               Verify data is correct (0's)
                                        ;
                                        ;               Set RAR to 44
                                        ;               Execute 24. JMAP's
                                        ;               Read EBUF (Contains contents of Reg 17)
                                        ;               Verify data is correct (0's)
                                        ;
                                        ;               Set RAR to 73
                                        ;               Execute 14. JMAP's
                                        ;               Read EBUF (Contains contents of Reg 0 and Q)
                                        ;               Verify data is correct (1's)
                                        ;
                                        ; Failure:      ---
                                        ;#******************************************************************

                                        ; Test data

003130' 254 00 0 00 003141'             TSTA20: JRST    TG20                    ; go start test
003131' 420402  000020                          EBUS!ALU!NDMP!ZALU!20           ; test mask
003132' 003174' 007145'                         T20M,,[ASCIZ ^2901 NOT_R_AND_S Test^]
003133' 006620' 000000                          [EXP MLAST!E23],,0
003134' 000000  003421'                         TSTA21                          ; failure test table
003135' 000000  003764'                         TSTA22                          ;   ...
003136' 000000  004445'                         TSTA23
003137' 000000  004752'                         TSTA24
003140' 777777  777777                          -1

                                        ; Start test

003141' 201 00 0 00 000000'             TG20:   MOVEI   Z4                      ; get address of module start
003142' 260 17 0 00 002665*                     GO      TRACE                   ; handle trace output
003143' 201 01 0 00 003174'                     MOVEI   1,T20M                  ; set up microcode address
003144' 260 17 0 00 002667*                     GO      TLOAD                   ; load/verify it
003145' 263 17 0 00 000000                      RTN                             ; failed - exit test
```

```
                                 ; Initialization

003146' 400 15 0 00 000000      TL20:   SETZ    ERFLG,                  ; clear error flag
003147' 260 17 0 00 002672*             GO      IPACLR                  ; clear port
003150' 402 00 0 00 002673*             SETZM   TSTSUB                  ; initialize subtest number
003151' 201 06 0 00 003163'             MOVEI   6,TS20                  ; get sstep table address

                                 ; Loop on single step table entries

003152' 260 17 0 00 002675*     TA20:   GO      BEXEC                   ; execute table entry
003153' 254 00 0 00 003162'             JRST    TX20                    ; end of sstep table
003154' 254 00 0 00 003152'             JRST    TA20                    ; keep looping after call
003155' 474 15 0 00 000000              SETO    ERFLG,                  ; error - EBUF data incorrect

                                 ; Handle error printouts and scope looping

003156' 027 00 0 00 003172'             SCOPER  MA20                    ; print error message
003157' 254 00 0 00 003146'             JRST    TL20                    ; loop on error
003160' 254 00 0 00 003162'             JRST    TX20                    ; altmode exit
003161' 322 15 0 00 003152'             JUMPE   ERFLG,TA20              ; do next sstep table entry

                                 ; End of test

003162' 263 17 0 00 000000      TX20:   RTN                             ; return

                                 ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

003163' 104400  000043          TS20:   ATABLE  (SSSTRT,^D36,0,43,0)
003164' 000000  000000
003165' 103000  440073                  ATABLE  (SSSTRT,^D24,44,73,0)
003166' 000000  000000
003167' 101600  740111                  ATABLE  (SSSTRT,^D14,74,111,-1)
003170' 777777  777777
003171' 000000  000000                  ATABLE  (SSLAST)

                                 ; Error messages

003172' 140000  007152'         MA20:   MSG!TXNOT![ASCIZ /2901 not doing an NAND function properly (result in EBUF)/]
003173' 000000000000#                   LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                 ; Microcode:     Verify that 0's NOT_R_AND_S 0's ==> 0's

003174' 000000  010000          T20M:   MWORD   <ADDR=0,JMAP,J=1,SDO,AND,D=3>           ; 0
003175' 743000  000040
003176' 000100  000000                  MWORD   <CONT,B=17,SDO,AND,D=3>                 ; 1
003177' 743007  400340
003200' 000200  000000                  MWORD   <CONT,SDO,AND,D=0>                      ; 2
003201' 740000  000340
003202' 000300  000000                  MWORD   <CONT,B=1,SAQ,NAND,D=2,CRY>             ; 3
003203' 052000  400740
003204' 000400  000000                  MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 4
003205' 132017  400340
003206' 000500  000000                  MWORD   <CONT,B=1,SAQ,NAND,D=3,CRY>             ; 5
003207' 053000  400740
```

```
2721  003210' 000600  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 6
2722  003211' 132017  400340
2723  003212' 000700  000000          MWORD   <CONT,B=1,SAQ,NAND,D=5,CRY>             ; 7
2724  003213' 055000  400740
2725  003214' 001000  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 10
2726  003215' 132017  400340
2727  003216' 001100  000000          MWORD   <CONT,B=1,SAQ,NAND,D=7,CRY>             ; 11
2728  003217' 057000  400740
2729  003220' 001200  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 12
2730  003221' 132017  400340
2731  003222' 001300  000000          MWORD   <CONT,B=1,SAB,NAND,D=2>                 ; 13
2732  003223' 152000  400340
2733  003224' 001400  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 14
2734  003225' 132017  400340
2735  003226' 001500  000000          MWORD   <CONT,B=1,SAB,NAND,D=3>                 ; 15
2736  003227' 153000  400340
2737  003230' 001600  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 16
2738  003231' 132017  400340
2739  003232' 001700  000000          MWORD   <CONT,B=1,SAB,NAND,D=5,CRY>             ; 17
2740  003233' 155000  400740
2741  003234' 002000  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 20
2742  003235' 132017  400340
2743  003236' 002100  000000          MWORD   <CONT,B=1,SAB,NAND,D=7>                 ; 21
2744  003237' 157000  400340
2745  003240' 002200  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 22
2746  003241' 132017  400340
2747  003242' 002300  000000          MWORD   <CONT,B=1,SOQ,NAND,D=2,CRY>             ; 23
2748  003243' 252000  400740
2749  003244' 002400  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 24
2750  003245' 132017  400340
2751  003246' 002500  000000          MWORD   <CONT,B=1,SOQ,NAND,D=3>                 ; 25
2752  003247' 253000  400340
2753  003250' 002600  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 25
2754  003251' 132017  400340
2755  003252' 002700  000000          MWORD   <CONT,B=1,SOQ,NAND,D=5>                 ; 27
2756  003253' 255000  400340
2757  003254' 003000  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 30
2758  003255' 132017  400340
2759  003256' 003100  000000          MWORD   <CONT,B=1,SOQ,NAND,D=7,CRY>             ; 31
2760  003257' 257000  400740
2761  003260' 003200  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 32
2762  003261' 132017  400340
2763  003262' 003300  000000          MWORD   <CONT,B=1,SOA,NAND,D=2>                 ; 33
2764  003263' 452000  400340
2765  003264' 003400  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 34
2766  003265' 132017  400340
2767  003266' 003500  000000          MWORD   <CONT,B=1,SOA,NAND,D=3,CRY>             ; 35
2768  003267' 453000  400740
2769  003270' 003600  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 36
2770  003271' 132017  400340
2771  003272' 003700  000000          MWORD   <CONT,B=1,SOA,NAND,D=5>                 ; 37
2772  003273' 455000  400340
2773  003274' 004000  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 40
2774  003275' 132017  400340
2775  003276' 004100  000000          MWORD   <CONT,B=1,SOA,NAND,D=7>                 ; 41
```

```
2776  003277' 457000  400340          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>                      ; 42
2777  003300' 004200  000000
2778  003301' 132017  400340          MWORD   <JMAP,J=43,A=17,SOA,OR,D=1,SELE,MGC=4,OENA> ; 43
2779  003302' 004300  432004
2780  003303' 431170  005040
2781
2782                          ;       Verify that 1's NOT_R_AND_S 1's ==> 0's
2783
2784  003304' 004400  450000          MWORD   <JMAP,J=45,SDO,AND,D=3>                         ; 44
2785  003305' 743000  000040
2786  003306' 004500  000000          MWORD   <CONT,SOA,XNOR,D=0>                             ; 45
2787  003307' 470000  000340
2788  003310' 004600  000000          MWORD   <CONT,SOA,XNOR,D=2>                             ; 46
2789  003311' 472000  000340
2790  003312' 004700  000000          MWORD   <CONT,B=1,SAQ,NAND,D=2>                         ; 47
2791  003313' 052000  400340
2792  003314' 005000  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>                      ; 50
2793  003315' 132017  400340
2794  003316' 005100  000000          MWORD   <CONT,B=1,SAQ,NAND,D=3>                         ; 51
2795  003317' 053000  400340
2796  003320' 005200  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>                      ; 52
2797  003321' 132017  400340
2798  003322' 005300  000000          MWORD   <CONT,B=1,SAQ,NAND,D=5>                         ; 53
2799  003323' 055000  400340
2800  003324' 005400  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>                      ; 54
2801  003325' 132017  400340
2802  003326' 005500  000000          MWORD   <CONT,B=1,SAQ,NAND,D=7,CRY>                     ; 55
2803  003327' 057000  400740
2804  003330' 005600  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>                      ; 56
2805  003331' 132017  400340
2806  003332' 005700  000000          MWORD   <CONT,B=1,SOA,OR,D=2>                           ; 57
2807  003333' 432000  400340
2808  003334' 006000  000000          MWORD   <CONT,B=1,SAB,NAND,D=2,CRY>                     ; 60
2809  003335' 152000  400740
2810  003336' 006100  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>                      ; 61
2811  003337' 132017  400340
2812  003340' 006200  000000          MWORD   <CONT,B=1,SOA,OR,D=2>                           ; 62
2813  003341' 432000  400340
2814  003342' 006300  000000          MWORD   <CONT,B=1,SAB,NAND,D=3>                         ; 63
2815  003343' 153000  400340
2816  003344' 006400  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>                      ; 64
2817  003345' 132017  400340
2818  003346' 006500  000000          MWORD   <CONT,B=1,SOA,OR,D=2>                           ; 65
2819  003347' 432000  400340
2820  003350' 006600  000000          MWORD   <CONT,B=1,SAB,NAND,D=5>                         ; 66
2821  003351' 155000  400340
2822  003352' 006700  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>                      ; 67
2823  003353' 132017  400340
2824  003354' 007000  000000          MWORD   <CONT,B=1,SOA,OR,D=2>                           ; 70
2825  003355' 432000  400340
2826  003356' 007100  000000          MWORD   <CONT,B=1,SAB,NAND,D=7,CRY>                     ; 71
2827  003357' 157000  400740
2828  003360' 007200  000000          MWORD   <CONT,A=1,B=17,SAB,OR,D=2>                      ; 72
2829  003361' 132017  400340
2830  003362' 007300  732004          MWORD   <JMAP,J=73,A=17,SOA,OR,D=1,SELE,MGC=4,OENA> ; 73
```

```
003363' 431170  005040

                                        ;       Verify that 0's NOT_R_AND_S 1's ==> 1's

003364' 007400  750000                          MWORD   <JMAP,J=75,SD0,AND,D=3>                         ; 74
003365' 743000  000040
003366' 007500  000000                          MWORD   <CONT,SOA,XNOR,D=0>                             ; 75
003367' 470000  000340
003370' 007600  000000                          MWORD   <CONT,B=1,SOA,XNOR,D=2>                         ; 76
003371' 472000  400340
003372' 007700  000000                          MWORD   <CONT,SAQ,NAND,D=0>                             ; 77
003373' 050000  000340
003374' 010000  000000                          MWORD   <CONT,SAQ,NAND,D=1,CRY>                         ; 100
003375' 051000  000740
003376' 010100  000000                          MWORD   <CONT,SAQ,NAND,D=2,CRY>                         ; 101
003377' 052000  000740
003400' 010200  000000                          MWORD   <CONT,B=1,SAB,NAND,D=2,CRY>                     ; 102
003401' 152000  400740
003402' 010300  000000                          MWORD   <CONT,B=1,SAB,NAND,D=3,CRY>                     ; 103
003403' 153000  400740
003404' 010400  000000                          MWORD   <CONT,SOQ,NAND,D=0>                             ; 104
003405' 250000  000340
003406' 010500  000000                          MWORD   <CONT,B=1,SOB,NAND,D=2>                         ; 105
003407' 352000  400340
003410' 010600  000000                          MWORD   <CONT,B=1,SOB,NAND,D=3>                         ; 106
003411' 353000  400340
003412' 010700  000000                          MWORD   <CONT,A=1,B=1,SOA,NAND,D=2>                     ; 107
003413' 452010  400340
003414' 011000  000000                          MWORD   <CONT,A=1,B=1,SOA,NAND,D=3>                     ; 110
003415' 453010  400340
003416' 011101  112004                          MWORD   <JMAP,J=111,SAQ,AND,D=1,SELE,MGC=4,OENA>; 111
003417' 041000  005040
003420' 777777  777777                          -1
```

```

                                        ;#*****************************************************************
                                        ;* Test 21 - 2901 XOR Test
                                        ;
                                        ; Description:  Verify that 1's XOR 1's ==> 0's
                                        ;               Verify that 0's XOR 0's ==> 0's
                                        ;               Verify that 0's XOR 1's ==> 1's
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load microcode
                                        ;
                                        ;               Set RAR to 0
                                        ;               Execute 25. JMAP's
                                        ;               Read EBUF (Contains contents of Reg 17)
                                        ;               Verify data is correct (0's)
                                        ;
                                        ;               Set RAR to 33
                                        ;               Execute 30. JMAP's
                                        ;               Read EBUF (Contains contents of Reg 17)
                                        ;               Verify data is correct (0's)
                                        ;
                                        ;               Set RAR to 67
                                        ;               Execute 40. JMAP's
                                        ;               Read EBUF (Contains contents of Reg 17)
                                        ;               Verify data is correct (1's)
                                        ;
                                        ; Failure:      ---
                                        ;#*****************************************************************

                                        ; Test data

 003421' 254 00 0 00 003432'            TSTA21: JRST    TG21                    ; go start test
 003422' 420402  000021                         EBUS!ALU!NDMP!ZALU!21           ; test mask
 003423' 003465' 007166'                        T21M,,[ASCIZ ^2901 XOR Test^]
 003424' 006620' 000000                         [EXP MLAST!E23],,0
 003425' 000000  003764'                        TSTA22                          ; failure test table
 003426' 000000  004445'                        TSTA23                          ;   ...
 003427' 000000  004752'                        TSTA24
 003430' 000000  005261'                        TSTA25
 003431' 777777  777777                         -1

                                        ; Start test

 003432' 201 00 0 00 000000'            TG21:   MOVEI   Z4                      ; get address of module start
 003433' 260 17 0 00 003142*                    GO      TRACE                   ; handle trace output
 003434' 201 01 0 00 003465'                    MOVEI   1,T21M                  ; set up microcode address
 003435' 260 17 0 00 003144*                    GO      TLOAD                   ; load/verify it
 003436' 263 17 0 00 000000                     RTN                             ; failed - exit test
```

```

                                        ; Initialization

003437' 400 15 0 00 000000      TL21:   SETZ    ERFLG,                  ; clear error flag
003440' 260 17 0 00 003147*             GO      IPACLR                  ; clear port
003441' 402 00 0 00 003150*             SETZM   TSTSUB                  ; initialize subtest number
003442' 201 06 0 00 003454'             MOVEI   6,TS21                  ; get sstep table address

                                        ; Loop on single step table entries

003443' 260 17 0 00 003152*     TA21:   GO      BEXEC                   ; execute table entry
003444' 254 00 0 00 003453'             JRST    TX21                    ; end of sstep table
003445' 254 00 0 00 003443'             JRST    TA21                    ; keep looping after call
003446' 474 15 0 00 000000              SETO    ERFLG,                  ; error - EBUF data incorrect

                                        ; Handle error printouts and scope looping

003447' 027 00 0 00 003463'             SCOPER  MA21                    ; print error message
003450' 254 00 0 00 003437'             JRST    TL21                    ; loop on error
003451' 254 00 0 00 003453'             JRST    TX21                    ; altmode exit
003452' 322 15 0 00 003443'             JUMPE   ERFLG,TA21              ; do next sstep table entry

                                        ; End of test

003453' 263 17 0 00 000000      TX21:   RTN                             ; return

                                        ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

003454' 103100  000030          TS21:   ATABLE  (SSSTRT,^D25,0,30,0)
003455' 000000  000000
003456' 103600  310066                  ATABLE  (SSSTRT,^D30,31,66,0)
003457' 000000  000000
003460' 105000  670136                  ATABLE  (SSSTRT,^D40,67,136,-1)
003461' 777777  777777
003462' 000000  000000                  ATABLE  (SSLAST)

                                        ; Error messages

003463' 140000  007171'         MA21:   MSG!TXNOT![ASCIZ /2901 not doing an XOR function properly (result in EBUF)/]
003464' 000000000000#                   LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                        ; Microcode:    Verify that 1's XOR 1's ==> 0's

003465' 000000  010000          T21M:   MWORD   <ADDR=0,JMAP,J=1,SDO,AND,D=3>           ; 0
003466' 743000  000040
003467' 000100  000000                  MWORD   <CONT,B=17,SDO,AND,D=3>                 ; 1
003470' 743007  400340
003471' 000200  000000                  MWORD   <CONT,SOA,XNOR,D=0>                     ; 2
003472' 470000  000340
003473' 000300  000000                  MWORD   <CONT,SOA,XNOR,D=2>                     ; 3
003474' 472000  000340
003475' 000400  000000                  MWORD   <CONT,B=1,SAQ,XOR,D=2>                  ; 4
003476' 062000  400340
003477' 000500  000000                  MWORD   <CONT,A=1,B=17,SAB,OR,D=2>              ; 5
003500' 132017  400340
```

```
2967	003501'	000600	000000		MWORD	<CONT,B=1,SAQ,XOR,D=3,CRY>		; 6
2968	003502'	063000	400740
2969	003503'	000700	000000		MWORD	<CONT,A=1,B=17,SAB,OR,D=2>		; 7
2970	003504'	132017	400340
2971	003505'	001000	000000		MWORD	<CONT,B=1,SAQ,XOR,D=5,CRY>		; 10
2972	003506'	065000	400740
2973	003507'	001100	000000		MWORD	<CONT,A=1,B=17,SAB,OR,D=2>		; 11
2974	003510'	132017	400340
2975	003511'	001200	000000		MWORD	<CONT,B=1,SAQ,XOR,D=7,CRY>		; 12
2976	003512'	067000	400740
2977	003513'	001300	000000		MWORD	<CONT,A=1,B=17,SAB,OR,D=2>		; 13
2978	003514'	132017	400340
2979	003515'	001400	000000		MWORD	<CONT,B=1,SOA,OR,D=2>			; 14
2980	003516'	432000	400340
2981	003517'	001500	000000		MWORD	<CONT,B=1,SAB,XOR,D=2,CRY>		; 15
2982	003520'	162000	400740
2983	003521'	001600	000000		MWORD	<CONT,A=1,B=17,SAB,OR,D=2>		; 16
2984	003522'	132017	400340
2985	003523'	001700	000000		MWORD	<CONT,B=1,SOA,OR,D=2>			; 17
2986	003524'	432000	400340
2987	003525'	002000	000000		MWORD	<CONT,B=1,SAB,XOR,D=3>			; 20
2988	003526'	163000	400340
2989	003527'	002100	000000		MWORD	<CONT,A=1,B=17,SAB,OR,D=2>		; 21
2990	003530'	132017	400340
2991	003531'	002200	000000		MWORD	<CONT,B=1,SOA,OR,D=2>			; 22
2992	003532'	432000	400340
2993	003533'	002300	000000		MWORD	<CONT,B=1,SAB,XOR,D=5>			; 23
2994	003534'	165000	400340
2995	003535'	002400	000000		MWORD	<CONT,A=1,B=17,SAB,OR,D=2>		; 24
2996	003536'	132017	400340
2997	003537'	002500	000000		MWORD	<CONT,B=1,SOA,OR,D=2>			; 25
2998	003540'	432000	400340
2999	003541'	002600	000000		MWORD	<CONT,B=1,SAB,XOR,D=7>			; 26
3000	003542'	167000	400340
3001	003543'	002700	000000		MWORD	<CONT,A=1,B=17,SAB,OR,D=2>		; 27
3002	003544'	132017	400340
3003	003545'	003000	302004		MWORD	<JMAP,J=30,A=17,SOA,OR,D=1,SELE,MGC=4,OENA> ; 30
3004	003546'	431170	005040
3005
3006						;	Verify that 0's XOR 0's ==> 0's
3007
3008	003547'	003100	320000		MWORD	<JMAP,J=32,SDO,AND,D=3>			; 31
3009	003550'	743000	000040
3010	003551'	003200	000000		MWORD	<CONT,SDO,AND,D=0>			; 32
3011	003552'	740000	000340
3012	003553'	003300	000000		MWORD	<CONT,SAQ,XOR,D=0>			; 33
3013	003554'	060000	000340
3014	003555'	003400	000000		MWORD	<CONT,SAQ,XOR,D=1,CRY>			; 34
3015	003556'	061000	000740
3016	003557'	003500	000000		MWORD	<CONT,SAQ,XOR,D=2,CRY>			; 35
3017	003560'	062000	000740
3018	003561'	003600	000000		MWORD	<CONT,SAQ,XOR,D=3,CRY>			; 36
3019	003562'	063000	000740
3020	003563'	003700	000000		MWORD	<CONT,SAQ,XOR,D=4>			; 37
3021	003564'	064000	000340
```

```
003565' 004000  000000                          MWORD   <CONT,SAQ,XOR,D=5,CRY>                 ; 40
003566' 065000  000740
003567' 004100  000000                          MWORD   <CONT,SAQ,XOR,D=6>                     ; 41
003570' 066000  000340
003571' 004200  000000                          MWORD   <CONT,SAQ,XOR,D=7>                     ; 42
003572' 067000  000340
003573' 004300  000000                          MWORD   <CONT,SAB,XOR,D=1>                     ; 43
003574' 161000  000340
003575' 004400  000000                          MWORD   <CONT,SAB,XOR,D=2>                     ; 44
003576' 162000  000340
003577' 004500  000000                          MWORD   <CONT,SAB,XOR,D=3,CRY>                 ; 45
003600' 163000  000740
003601' 004600  000000                          MWORD   <CONT,SAB,XOR,D=5>                     ; 46
003602' 165000  000340
003603' 004700  000000                          MWORD   <CONT,SAB,XOR,D=7>                     ; 47
003604' 167000  000340
003605' 005000  000000                          MWORD   <CONT,B=1,SOQ,XOR,D=0>                 ; 50
003606' 260000  400340
003607' 005100  000000                          MWORD   <CONT,B=1,SOQ,XOR,D=1>                 ; 51
003610' 261000  400340
003611' 005200  000000                          MWORD   <CONT,B=1,SOQ,XOR,D=4,CRY>             ; 52
003612' 264000  400740
003613' 005300  000000                          MWORD   <CONT,B=1,SOQ,XOR,D=6>                 ; 53
003614' 266000  400340
003615' 005400  000000                          MWORD   <CONT,SOA,XOR,D=1>                     ; 54
003616' 461000  000340
003617' 005500  000000                          MWORD   <CONT,SOA,XOR,D=2>                     ; 55
003620' 462000  000340
003621' 005600  000000                          MWORD   <CONT,SOA,XOR,D=3,CRY>                 ; 56
003622' 463000  000740
003623' 005700  000000                          MWORD   <CONT,SOA,XOR,D=5,CRY>                 ; 57
003624' 465000  000740
003625' 006000  000000                          MWORD   <CONT,SOA,XOR,D=7>                     ; 60
003626' 467000  000340
003627' 006100  000000                          MWORD   <CONT,SOB,XOR,D=1>                     ; 61
003630' 361000  000340
003631' 006200  000000                          MWORD   <CONT,SOB,XOR,D=2>                     ; 62
003632' 362000  000340
003633' 006300  000000                          MWORD   <CONT,SOB,XOR,D=3>                     ; 63
003634' 363000  000340
003635' 006400  000000                          MWORD   <CONT,SOB,XOR,D=5,CRY>                 ; 64
003636' 365000  000740
003637' 006500  000000                          MWORD   <CONT,SOB,XOR,D=7>                     ; 65
003640' 367000  000340
003641' 006600  662004                          MWORD   <JMAP,J=66,SAQ,OR,D=1,SELE,MGC=4,OENA> ; 66
003642' 031000  005040

                                        ;       Verify that 0's XOR 1's ==> 1's

003643' 006700  700000                          MWORD   <JMAP,J=70,SDO,AND,D=3>                ; 67
003644' 743000  000040
003645' 007000  000000                          MWORD   <CONT,B=1,SDO,AND,D=3>                 ; 70
003646' 743000  400340
003647' 007100  000000                          MWORD   <CONT,SOA,XNOR,D=0>                    ; 71
003650' 470000  000340
```

```
 3077  003651' 007200  000000          MWORD   <CONT,B=2,SOA,XNOR,D=2>                 ; 72
 3078  003652' 472001  000340
 3079  003653' 007300  000000          MWORD   <CONT,B=17,SOA,XNOR,D=2>                ; 73
 3080  003654' 472007  400340
 3081  003655' 007400  000000          MWORD   <CONT,B=3,SOQ,XOR,D=2,CRY>              ; 74
 3082  003656' 262001  400740
 3083  003657' 007500  000000          MWORD   <CONT,A=3,B=17,SAB,AND,D=2>             ; 75
 3084  003660' 142037  400340
 3085  003661' 007600  000000          MWORD   <CONT,B=3,SOQ,XOR,D=3,CRY>              ; 76
 3086  003662' 263001  400740
 3087  003663' 007700  000000          MWORD   <CONT,A=3,B=17,SAB,AND,D=2>             ; 77
 3088  003664' 142037  400340
 3089  003665' 010000  000000          MWORD   <CONT,A=2,B=3,SOA,XOR,D=2,CRY>          ; 100
 3090  003666' 462021  400740
 3091  003667' 010100  000000          MWORD   <CONT,A=3,B=17,SAB,AND,D=2>             ; 101
 3092  003670' 142037  400340
 3093  003671' 010200  000000          MWORD   <CONT,A=2,B=3,SOA,XOR,D=3,CRY>          ; 102
 3094  003672' 463021  400740
 3095  003673' 010300  000000          MWORD   <CONT,A=3,B=17,SAB,AND,D=2>             ; 103
 3096  003674' 142037  400340
 3097  003675' 010400  000000          MWORD   <CONT,A=2,B=3,SOA,OR,D=2>               ; 104
 3098  003676' 432021  400340
 3099  003677' 010500  000000          MWORD   <CONT,B=3,SOB,XOR,D=2>                  ; 105
 3100  003700' 362001  400340
 3101  003701' 010600  000000          MWORD   <CONT,A=3,B=17,SAB,AND,D=2>             ; 106
 3102  003702' 142037  400340
 3103  003703' 010700  000000          MWORD   <CONT,A=2,B=3,SOA,OR,D=2>               ; 107
 3104  003704' 432021  400340
 3105  003705' 011000  000000          MWORD   <CONT,B=3,SOB,XOR,D=3,CRY>              ; 110
 3106  003706' 363001  400740
 3107  003707' 011100  000000          MWORD   <CONT,A=3,B=17,SAB,AND,D=2>             ; 111
 3108  003710' 142037  400340
 3109  003711' 011200  000000          MWORD   <CONT,A=1,B=3,SAQ,XOR,D=2>              ; 112
 3110  003712' 062011  400340
 3111  003713' 011300  000000          MWORD   <CONT,A=3,B=17,SAB,AND,D=2>             ; 113
 3112  003714' 142037  400340
 3113  003715' 011400  000000          MWORD   <CONT,A=1,B=3,SAQ,XOR,D=3>              ; 114
 3114  003716' 063011  400340
 3115  003717' 011500  000000          MWORD   <CONT,A=3,B=17,SAB,AND,D=2>             ; 115
 3116  003720' 142037  400340
 3117  003721' 011600  000000          MWORD   <CONT,A=1,B=3,SAQ,XOR,D=5,CRY>          ; 116
 3118  003722' 065011  400740
 3119  003723' 011700  000000          MWORD   <CONT,A=3,B=17,SAB,AND,D=2>             ; 117
 3120  003724' 142037  400340
 3121  003725' 012000  000000          MWORD   <CONT,A=1,B=3,SAQ,XOR,D=7,CRY>          ; 120
 3122  003726' 067011  400740
 3123  003727' 012100  000000          MWORD   <CONT,A=3,B=17,SAB,AND,D=2>             ; 121
 3124  003730' 142037  400340
 3125  003731' 012200  000000          MWORD   <CONT,A=2,B=3,SOA,OR,D=2>               ; 122
 3126  003732' 432021  400340
 3127  003733' 012300  000000          MWORD   <CONT,B=3,SAB,XOR,D=2>                  ; 123
 3128  003734' 162001  400340
 3129  003735' 012400  000000          MWORD   <CONT,A=3,B=17,SAB,AND,D=2>             ; 124
 3130  003736' 142037  400340
 3131  003737' 012500  000000          MWORD   <CONT,A=2,B=3,SOA,OR,D=2>               ; 125
```

```
003740' 432021  400340        MWORD   <CONT,B=3,SAB,XOR,D=3,CRY>                   ; 126
003741' 012600  000000
003742' 163001  400740        MWORD   <CONT,A=3,B=17,SAB,AND,D=2>                  ; 127
003743' 012700  000000
003744' 142037  400340        MWORD   <CONT,A=2,B=3,SOA,OR,D=2>                    ; 130
003745' 013000  000000
003746' 432021  400340        MWORD   <CONT,B=3,SAB,XOR,D=5,CRY>                   ; 131
003747' 013100  000000
003750' 165001  400740        MWORD   <CONT,A=3,B=17,SAB,AND,D=2>                  ; 132
003751' 013200  000000
003752' 142037  400340        MWORD   <CONT,A=2,B=3,SOA,OR,D=2>                    ; 133
003753' 013300  000000
003754' 432021  400340        MWORD   <CONT,B=3,SAB,XOR,D=7>                       ; 134
003755' 013400  000000
003756' 167001  400340        MWORD   <CONT,A=3,B=17,SAB,AND,D=2>                  ; 135
003757' 013500  000000
003760' 142037  400340        MWORD   <JMAP,J=136,SOQ,OR,D=1,SELE,MGC=4,OENA> ; 136
003761' 013601  362004
003762' 231000  005040
003763' 777777  777777        -1
```

```

                                    ;#*********************************************************************
                                    ;* Test 22 - 2901 XNOR Test
                                    ;
                                    ; Description:  Verify that 0's XNOR 1's ==> 0's
                                    ;               Verify that 0's XNOR 0's ==> 1's
                                    ;               Verify that 1's XNOR 1's ==> 1's
                                    ;
                                    ; Procedure:    Clear Port
                                    ;               Load microcode/set start address 0
                                    ;
                                    ;               Set RAR to 0
                                    ;               Execute 49. JMAP's
                                    ;               Read EBUF (Contains contents of Reg 17)
                                    ;               Verify data is correct (0's)
                                    ;
                                    ;               Set RAR to 47
                                    ;               Execute 28. JMAP's
                                    ;               Read EBUF (Contains contents of Reg 17)
                                    ;               Verify data is correct (1's)
                                    ;
                                    ;               Set RAR to 103
                                    ;               Execute 67. JMAP's
                                    ;               Read EBUF (Contains contents of Reg 17)
                                    ;               Verify data is correct (1's)
                                    ;
                                    ; Failure:      ---
                                    ;#*********************************************************************

                                    ; Test data

003764' 254 00 0 00 003775'         TSTA22: JRST    TG22                    ; go start test
003765' 420402  000022                      EBUS!ALU!NDMP!ZALU!22           ; test mask
003766' 004030' 007205'                     T22M,,[ASCIZ ^2901 XNOR Test^]
003767' 006620' 000000                      [EXP MLAST!E23],,0
003770' 000000  004445'                     TSTA23                          ; failure test table
003771' 000000  004752'                     TSTA24                          ;   ...
003772' 000000  005261'                     TSTA25
003773' 000000  005552'                     TSTA26
003774' 777777  777777                      -1

                                    ; Start test

003775' 201 00 0 00 000000'         TG22:   MOVEI   Z4                      ; get address of module start
003776' 260 17 0 00 003433*                 GO      TRACE                   ; handle trace output
003777' 201 01 0 00 004030'                 MOVEI   1,T22M                  ; set up microcode address
004000' 260 17 0 00 003435*                 GO      TLOAD                   ; load/verify it
004001' 263 17 0 00 000000                  RTN                             ; failed - exit test
```

```

                                         ; Initialization

004002' 400 15 0 00 000000               TL22:   SETZ    ERFLG,                  ; clear error flag
004003' 260 17 0 00 003440*                      GO      IPACLR                  ; clear port
004004' 402 00 0 00 003441*                      SETZM   TSTSUB                  ; initialize subtest number
004005' 201 06 0 00 004017'                      MOVEI   6,TS22                  ; get sstep table address

                                         ; Loop on single step table entries

004006' 260 17 0 00 003443*              TA22:   GO      BEXEC                   ; execute table entry
004007' 254 00 0 00 004016'                      JRST    TX22                    ; end of sstep table
004010' 254 00 0 00 004006'                      JRST    TA22                    ; keep looping after call
004011' 474 15 0 00 000000                       SETO    ERFLG,                  ; error - EBUF data incorrect

                                         ; Handle error printouts and scope looping

004012' 027 00 0 00 004026'                      SCOPER  MA22                    ; print error message
004013' 254 00 0 00 004002'                      JRST    TL22                    ; loop on error
004014' 254 00 0 00 004016'                      JRST    TX22                    ; altmode exit
004015' 322 15 0 00 004006'                      JUMPE   ERFLG,TA22              ; do next sstep table entry

                                         ; End of test

004016' 263 17 0 00 000000               TX22:   RTN                             ; return

                                         ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

004017' 104700  000046                   TS22:   ATABLE  (SSSTRT,^D39,0,46,0)
004020' 000000  000000
004021' 104400  470102                           ATABLE  (SSSTRT,^D28,47,102,-1)
004022' 777777  777777
004023' 110301  030205                           ATABLE  (SSSTRT,^D67,103,205,-1)
004024' 777777  777777
004025' 000000  000000                           ATABLE  (SSLAST)

                                         ; Error messages

004026' 140000  007210'                  MA22:   MSG!TXNOT![ASCIZ /2901 not doing an XNOR function properly (result in EBUF)/]
004027' 000000000000#                            LAST!CALL!TXALL!BBPNT           ; print sequencer data

                                         ; Microcode:    Verify that 0's XNOR 1's ==> 0's

004030' 000000  010000                   T22M:   MWORD   <ADDR=0,JMAP,J=1,SDO,AND,D=3>           ; 0 R0 <== 0's
004031' 743000  000040
004032' 000100  000000                           MWORD   <CONT,B=1,SOA,XNOR,D=2>                 ; 1 R1 <== 1's
004033' 472000  400340
004034' 000200  000000                           MWORD   <CONT,SOA,XNOR,D=0>                     ; 2 R2 <== 1's
004035' 470000  000340

004036' 000300  000000                           MWORD   <CONT,SAQ,XNOR,D=1,CRY>                 ; 3
004037' 071000  000740
004040' 000400  000000                           MWORD   <CONT,SAQ,XNOR,D=2,CRY>                 ; 4
004041' 072000  000740
004042' 000500  000000                           MWORD   <CONT,SAQ,XNOR,D=3,CRY>                 ; 5
```

```
3255  004043' 073000  000740
3256  004044' 000600  000000                MWORD   <CONT,SAQ,XNOR,D=5,CRY>                 ; 6
3257  004045' 075000  000740
3258  004046' 000700  000000                MWORD   <CONT,SAQ,XNOR,D=7,CRY>                 ; 7
3259  004047' 077000  000740
3260  004050' 001000  000000                MWORD   <CONT,A=1,SAB,XNOR,D=1>                 ; 10
3261  004051' 171010  000340
3262  004052' 001100  000000                MWORD   <CONT,A=1,SAB,XNOR,D=2>                 ; 11
3263  004053' 172010  000340
3264  004054' 001200  000000                MWORD   <CONT,A=1,SAB,XNOR,D=3>                 ; 12
3265  004055' 173010  000340
3266  004056' 001300  000000                MWORD   <CONT,A=1,SAB,XNOR,D=5>                 ; 13
3267  004057' 175010  000340
3268  004060' 001400  000000                MWORD   <CONT,A=1,SAB,XNOR,D=7>                 ; 14
3269  004061' 177010  000340
3270  004062' 001500  000000                MWORD   <CONT,B=2,SOQ,XNOR,D=2>                 ; 15
3271  004063' 272001  000340
3272  004064' 001600  000000                MWORD   <CONT,A=2,SAB,OR,D=2>                   ; 16
3273  004065' 132020  000340
3274  004066' 001700  000000                MWORD   <CONT,B=2,SOQ,XNOR,D=3>                 ; 17
3275  004067' 273001  000340
3276  004070' 002000  000000                MWORD   <CONT,A=2,SAB,OR,D=2>                   ; 20
3277  004071' 132020  000340
3278  004072' 002100  000000                MWORD   <CONT,B=2,SOQ,XNOR,D=5>                 ; 21
3279  004073' 275001  000340
3280  004074' 002200  000000                MWORD   <CONT,A=2,SAB,OR,D=2>                   ; 22
3281  004075' 132020  000340
3282  004076' 002300  000000                MWORD   <CONT,B=2,SOQ,XNOR,D=7>                 ; 23
3283  004077' 277001  000340
3284  004100' 002400  000000                MWORD   <CONT,A=2,SAB,OR,D=2>                   ; 24
3285  004101' 132020  000340
3286  004102' 002500  000000                MWORD   <CONT,A=1,SOA,XNOR,D=1>                 ; 25
3287  004103' 471010  000340
3288  004104' 002600  000000                MWORD   <CONT,A=1,SOA,XNOR,D=2,CRY>             ; 26
3289  004105' 472010  000740
3290  004106' 002700  000000                MWORD   <CONT,A=1,SOA,XNOR,D=3,CRY>             ; 27
3291  004107' 473010  000740
3292  004110' 003000  000000                MWORD   <CONT,A=1,SOA,XNOR,D=5>                 ; 30
3293  004111' 475010  000340
3294  004112' 003100  000000                MWORD   <CONT,A=1,SOA,XNOR,D=7>                 ; 31
3295  004113' 477010  000340
3296  004114' 003200  000000                MWORD   <CONT,A=1,B=2,SOA,OR,D=2>               ; 32
3297  004115' 432011  000340
3298  004116' 003300  000000                MWORD   <CONT,B=2,SOB,XNOR,D=2>                 ; 33
3299  004117' 372001  000340
3300  004120' 003400  000000                MWORD   <CONT,A=2,SAB,OR,D=2>                   ; 34
3301  004121' 132020  000340
3302  004122' 003500  000000                MWORD   <CONT,A=1,B=2,SOA,OR,D=2>               ; 35
3303  004123' 432011  000340
3304  004124' 003600  000000                MWORD   <CONT,B=2,SOB,XNOR,D=3>                 ; 36
3305  004125' 373001  000340
3306  004126' 003700  000000                MWORD   <CONT,A=2,SAB,OR,D=2>                   ; 37
3307  004127' 132020  000340
3308  004130' 004000  000000                MWORD   <CONT,A=1,B=2,SOA,OR,D=2>               ; 40
3309  004131' 432011  000340
```

```
 3310  004132' 004100  000000          MWORD   <CONT,B=2,SOB,XNOR,D=5>                      ; 41
 3311  004133' 375001  000340
 3312  004134' 004200  000000          MWORD   <CONT,A=2,SAB,OR,D=2>                        ; 42
 3313  004135' 132020  000340
 3314  004136' 004300  000000          MWORD   <CONT,A=1,B=2,SOA,OR,D=2>                    ; 43
 3315  004137' 432011  000340
 3316  004140' 004400  000000          MWORD   <CONT,B=2,SOB,XNOR,D=7,CRY>                  ; 44
 3317  004141' 377001  000740
 3318  004142' 004500  000000          MWORD   <CONT,A=2,SAB,OR,D=2>                        ; 45
 3319  004143' 132020  000340
 3320  004144' 004600  462004          MWORD   <JMAP,J=46,SOA,OR,D=1,SELE,MGC=4,OENA>       ; 46
 3321  004145' 431000  005040
 3322
 3323                          ;       Verify that 0's XNOR 0's ==> 1's
 3324
 3325  004146' 004700  500000          MWORD   <JMAP,J=50,SDO,AND,D=3>                      ; 47 zero R0
 3326  004147' 743000  000040
 3327  004150' 005000  000000          MWORD   <CONT,SDO,AND,D=0>                           ; 50 zero Q
 3328  004151' 740000  000340
 3329  004152' 005100  000000          MWORD   <CONT,B=17,SOA,XNOR,D=2>                     ; 51 one's to R17
 3330  004153' 472007  400340
 3331  004154' 005200  000000          MWORD   <CONT,B=1,SAQ,XNOR,D=2>                      ; 52
 3332  004155' 072000  400340
 3333  004156' 005300  000000          MWORD   <CONT,A=1,B=17,SAB,AND,D=2>                  ; 53
 3334  004157' 142017  400340
 3335  004160' 005400  000000          MWORD   <CONT,B=1,SAQ,XNOR,D=3,CRY>                  ; 54
 3336  004161' 073000  400740
 3337  004162' 005500  000000          MWORD   <CONT,A=1,B=17,SAB,AND,D=2,CRY>              ; 55
 3338  004163' 142017  400740
 3339  004164' 005600  000000          MWORD   <CONT,B=1,SOB,AND,D=2,CRY>                   ; 56
 3340  004165' 342000  400740
 3341  004166' 005700  000000          MWORD   <CONT,B=1,SAB,XNOR,D=2,CRY>                  ; 57
 3342  004167' 172000  400740
 3343  004170' 006000  000000          MWORD   <CONT,A=1,B=17,SAB,AND,D=2,CRY>              ; 60
 3344  004171' 142017  400740
 3345  004172' 006100  000000          MWORD   <CONT,B=1,SOB,AND,D=2,CRY>                   ; 61
 3346  004173' 342000  400740
 3347  004174' 006200  000000          MWORD   <CONT,B=1,SAB,XNOR,D=3,CRY>                  ; 62
 3348  004175' 173000  400740
 3349  004176' 006300  000000          MWORD   <CONT,A=1,B=17,SAB,AND,D=2>                  ; 63
 3350  004177' 142017  400340
 3351  004200' 006400  000000          MWORD   <CONT,B=1,SOQ,XNOR,D=2>                      ; 64
 3352  004201' 272000  400340
 3353  004202' 006500  000000          MWORD   <CONT,A=1,B=17,SAB,AND,D=2>                  ; 65
 3354  004203' 142017  400340
 3355  004204' 006600  000000          MWORD   <CONT,B=1,SOQ,XNOR,D=3>                      ; 66
 3356  004205' 273000  400340
 3357  004206' 006700  000000          MWORD   <CONT,A=1,B=17,SAB,AND,D=2>                  ; 67
 3358  004207' 142017  400340
 3359  004210' 007000  000000          MWORD   <CONT,B=1,SOB,AND,D=2>                       ; 70
 3360  004211' 342000  400340
 3361  004212' 007100  000000          MWORD   <CONT,B=1,SOB,XNOR,D=2>                      ; 71
 3362  004213' 372000  400340
 3363  004214' 007200  000000          MWORD   <CONT,A=1,B=17,SAB,AND,D=2>                  ; 72
 3364  004215' 142017  400340
```

```
3365  004216' 007300  000000          MWORD   <CONT,B=1,SOB,AND,D=2>                   ; 73
3366  004217' 342000  400340
3367  004220' 007400  000000          MWORD   <CONT,B=1,SOB,XNOR,D=3>                  ; 74
3368  004221' 373000  400340
3369  004222' 007500  000000          MWORD   <CONT,A=1,B=17,SAB,AND,D=2>              ; 75
3370  004223' 142017  400340
3371  004224' 007600  000000          MWORD   <CONT,B=1,SOA,XNOR,D=2>                  ; 76
3372  004225' 472000  400340
3373  004226' 007700  000000          MWORD   <CONT,A=1,B=17,SAB,AND,D=2>              ; 77
3374  004227' 142017  400340
3375  004230' 010000  000000          MWORD   <CONT,B=1,SOA,XNOR,D=3>                  ; 100
3376  004231' 473000  400340
3377  004232' 010100  000000          MWORD   <CONT,A=1,B=17,SAB,AND,D=2>              ; 101
3378  004233' 142017  400340
3379  004234' 010201  032004          MWORD   <JMAP,J=103,A=17,SOA,OR,D=1,SELE,MGC=4,OENA> ; 102
3380  004235' 431170  005040
3381
3382                          ;       Verify that 1's XNOR 1's ==> 1's
3383
3384  004236' 010301  040000          MWORD   <JMAP,J=104,SDO,AND,D=3>                 ; 103
3385  004237' 743000  000040
3386  004240' 010400  000000          MWORD   <CONT,SAB,XNOR,D=2>                      ; 104
3387  004241' 172000  000340
3388  004242' 010500  000000          MWORD   <CONT,SOA,OR,B=1,D=2>                    ; 105
3389  004243' 432000  400340
3390  004244' 010600  000000          MWORD   <CONT,SOA,OR,B=2,D=2>                    ; 106
3391  004245' 432001  000340
3392  004246' 010700  000000          MWORD   <CONT,SOA,OR,B=3,D=2>                    ; 107
3393  004247' 432001  400340
3394  004250' 011000  000000          MWORD   <CONT,SOA,OR,B=4,D=2>                    ; 110
3395  004251' 432002  000340
3396  004252' 011100  000000          MWORD   <CONT,SOA,OR,B=5,D=2>                    ; 111
3397  004253' 432002  400340
3398  004254' 011200  000000          MWORD   <CONT,SOA,OR,B=6,D=2>                    ; 112
3399  004255' 432003  000340
3400  004256' 011300  000000          MWORD   <CONT,SOA,OR,B=7,D=2>                    ; 113
3401  004257' 432003  400340
3402  004260' 011400  000000          MWORD   <CONT,SOA,OR,B=10,D=2>                   ; 114
3403  004261' 432004  000340
3404  004262' 011500  000000          MWORD   <CONT,SOA,OR,B=11,D=2>                   ; 115
3405  004263' 432004  400340
3406  004264' 011600  000000          MWORD   <CONT,SOA,OR,B=12,D=2>                   ; 116
3407  004265' 432005  000340
3408  004266' 011700  000000          MWORD   <CONT,SOA,OR,B=13,D=2>                   ; 117
3409  004267' 432005  400340
3410  004270' 012000  000000          MWORD   <CONT,SOA,OR,B=14,D=2>                   ; 120
3411  004271' 432006  000340
3412  004272' 012100  000000          MWORD   <CONT,SOA,OR,B=15,D=2>                   ; 121
3413  004273' 432006  400340
3414  004274' 012200  000000          MWORD   <CONT,SOA,OR,B=16,D=2>                   ; 122
3415  004275' 432007  000340
3416  004276' 012300  000000          MWORD   <CONT,SOA,OR,B=17,D=2>                   ; 123
3417  004277' 432007  400340
3418  004300' 012400  000000          MWORD   <CONT,SOA,OR,D=0>                        ; 124
3419  004301' 430000  000340
```

```
3420  004302' 012500  000000          MWORD   <CONT,SAQ,XNOR,A=0,B=1,D=2>          ; 125
3421  004303' 072000  400340
3422  004304' 012600  000000          MWORD   <CONT,SAQ,XNOR,A=1,B=2,D=3>          ; 126
3423  004305' 073011  000340
3424  004306' 012700  000000          MWORD   <CONT,SAQ,XNOR,A=2,D=0>              ; 127
3425  004307' 070020  000340
3426  004310' 013000  000000          MWORD   <CONT,SAQ,XNOR,A=3,B=4,D=2>          ; 130
3427  004311' 072032  000340
3428  004312' 013100  000000          MWORD   <CONT,SAQ,XNOR,A=4,B=5,D=3>          ; 131
3429  004313' 073042  400340
3430  004314' 013200  000000          MWORD   <CONT,SAQ,XNOR,A=5,D=0>              ; 132
3431  004315' 070050  000340
3432  004316' 013300  000000          MWORD   <CONT,SAQ,XNOR,A=6,B=7,D=2>          ; 133
3433  004317' 072063  400340
3434  004320' 013400  000000          MWORD   <CONT,SAQ,XNOR,A=7,B=10,D=3>         ; 134
3435  004321' 073074  000340
3436  004322' 013500  000000          MWORD   <CONT,SAQ,XNOR,A=10,D=0>             ; 135
3437  004323' 070100  000340
3438  004324' 013600  000000          MWORD   <CONT,SAQ,XNOR,A=11,B=12,D=2>        ; 136
3439  004325' 072115  000340
3440  004326' 013700  000000          MWORD   <CONT,SAQ,XNOR,A=12,B=13,D=3>        ; 137
3441  004327' 073125  400340
3442  004330' 014000  000000          MWORD   <CONT,SAQ,XNOR,A=13,D=0>             ; 140
3443  004331' 070130  000340
3444  004332' 014100  000000          MWORD   <CONT,SAQ,XNOR,A=14,B=15,D=2>        ; 141
3445  004333' 072146  400340
3446  004334' 014200  000000          MWORD   <CONT,SAQ,XNOR,A=15,B=16,D=3>        ; 142
3447  004335' 073157  000340
3448  004336' 014300  000000          MWORD   <CONT,SAQ,XNOR,A=16,D=0>             ; 143
3449  004337' 070160  000340
3450  004340' 014400  000000          MWORD   <CONT,SAQ,XNOR,A=17,D=0>             ; 144
3451  004341' 070170  000340
3452  004342' 014500  000000          MWORD   <CONT,SAB,XNOR,A=0,B=1,D=2>          ; 145
3453  004343' 172000  400340
3454  004344' 014600  000000          MWORD   <CONT,SAB,XNOR,A=1,B=2,D=3>          ; 146
3455  004345' 173011  000340
3456  004346' 014700  000000          MWORD   <CONT,SAB,XNOR,A=2,B=3,D=2>          ; 147
3457  004347' 172021  400340
3458  004350' 015000  000000          MWORD   <CONT,SAB,XNOR,A=3,B=4,D=3>          ; 150
3459  004351' 173032  000340
3460  004352' 015100  000000          MWORD   <CONT,SAB,XNOR,A=4,B=5,D=2>          ; 151
3461  004353' 172042  400340
3462  004354' 015200  000000          MWORD   <CONT,SAB,XNOR,A=5,B=6,D=3>          ; 152
3463  004355' 173053  000340
3464  004356' 015300  000000          MWORD   <CONT,SAB,XNOR,A=6,B=7,D=2>          ; 153
3465  004357' 172063  400340
3466  004360' 015400  000000          MWORD   <CONT,SAB,XNOR,A=7,B=10,D=3>         ; 154
3467  004361' 173074  000340
3468  004362' 015500  000000          MWORD   <CONT,SAB,XNOR,A=10,B=11,D=2>        ; 155
3469  004363' 172104  400340
3470  004364' 015600  000000          MWORD   <CONT,SAB,XNOR,A=11,B=12,D=3>        ; 156
3471  004365' 173115  000340
3472  004366' 015700  000000          MWORD   <CONT,SAB,XNOR,A=12,B=13,D=2>        ; 157
3473  004367' 172125  400340
3474  004370' 016000  000000          MWORD   <CONT,SAB,XNOR,A=13,B=14,D=3>        ; 160
```

```
004371' 173136  000340
004372' 016100  000000            MWORD   <CONT,SAB,XNOR,A=14,B=15,D=2>           ; 161
004373' 172146  400340
004374' 016200  000000            MWORD   <CONT,SAB,XNOR,A=15,B=16,D=3>           ; 162
004375' 173157  000340
004376' 016300  000000            MWORD   <CONT,SAB,XNOR,A=16,B=17,D=2>           ; 163
004377' 172167  400340
004400' 016400  000000            MWORD   <CONT,SAB,XNOR,A=17,B=0,D=3>            ; 164
004401' 173170  000340
004402' 016500  000000            MWORD   <CONT,SAQ,AND,A=0,D=0>                  ; 165
004403' 040000  000340
004404' 016600  000000            MWORD   <CONT,SAQ,AND,A=1,D=0>                  ; 166
004405' 040010  000340
004406' 016700  000000            MWORD   <CONT,SAQ,AND,A=2,D=0>                  ; 167
004407' 040020  000340
004410' 017000  000000            MWORD   <CONT,SAQ,AND,A=3,D=0>                  ; 170
004411' 040030  000340
004412' 017100  000000            MWORD   <CONT,SAQ,AND,A=4,D=0>                  ; 171
004413' 040040  000340
004414' 017200  000000            MWORD   <CONT,SAQ,AND,A=5,D=0>                  ; 172
004415' 040050  000340
004416' 017300  000000            MWCRD   <CONT,SAQ,AND,A=6,D=0>                  ; 173
004417' 040060  000340
004420' 017400  000000            MWORD   <CONT,SAQ,AND,A=7,D=0>                  ; 174
004421' 040070  000340
004422' 017500  000000            MWORD   <CONT,SAQ,AND,A=10,D=0>                 ; 175
004423' 040100  000340
004424' 017600  000000            MWORD   <CONT,SAQ,AND,A=11,D=0>                 ; 176
004425' 040110  000340
004426' 017700  000000            MWORD   <CONT,SAQ,AND,A=12,D=0>                 ; 177
004427' 040120  000340
004430' 020000  000000            MWORD   <CONT,SAQ,AND,A=13,D=0>                 ; 200
004431' 040130  000340
004432' 020100  000000            MWORD   <CONT,SAQ,AND,A=14,D=0>                 ; 201
004433' 040140  000340
004434' 020200  000000            MWORD   <CONT,SAQ,AND,A=15,D=0>                 ; 202
004435' 040150  000340
004436' 020300  000000            MWORD   <CONT,SAQ,AND,A=16,D=0>                 ; 203
004437' 040160  000340
004440' 020400  000000            MWORD   <CONT,SAQ,AND,A=17,D=0>                 ; 204
004441' 040170  000340
004442' 020502  052004            MWORD   <JMAP,J=205,SOA,OR,D=1,SELE,MGC=4,OENA> ; 205
004443' 431000  005040
004444' 777777  777777            -1
```

```
                                   ;#*******************************************************************
                                   ;* Test 23 - 2901 R + S Test, No Carry
                                   ;
                                   ; Description:  Verify 0 + 0 => 0
                                   ;               Verify -1 + -1 => -2
                                   ;               Verify 0 + floating 1's ==> floating 1's
                                   ;               Verify 0 + floating 0's ==> floating 0's
                                   ;               Verify 1 + floating 0's ==> 11..11..00..00
                                   ;               Verify that float 1's + float 1's ==> float 1's
                                   ;                  (shifted left one bit)
                                   ;               Verify that float 1's + float 0's ==> -1
                                   ;
                                   ; Procedure:    Clear Port
                                   ;               Load microcode/set start address 0
                                   ;
                                   ;               Add 0 + 0, verify that this works
                                   ;               Add -1 + -1, verify that this works
                                   ;
                                   ;               Build a data pattern (floating 1)
                                   ;                 Add 0 + data pattern (floating 1)
                                   ;                 Add 0 + data pattern inverted (floating 0)
                                   ;                 Add 1 + data pattern inverted (floating 0)
                                   ;                 Add f1 + f1
                                   ;                 Add f1 + f0
                                   ;                 Verify that each works
                                   ;               Repeat with data patterns - floating 1's
                                   ;
                                   ; Failure:      ---
                                   ;#*******************************************************************

                                   ; Test data

004445' 254 00 0 00 004456'        TSTA23: JRST    TG23                    ; go start test
004446' 420402  000023                     EBUS!ALU!NDMP!ZALU!23           ; test mask
004447' 004605' 007224'                    T23M,,[ASCIZ ^2901 R + S Test - No Carry^]
004450' 006620' 000000                     [EXP MLAST!E23],,0
004451' 000000  004752'                    TSTA24                          ; failure test table
004452' 000000  005261'                    TSTA25                          ;   ...
004453' 000000  005552'                    TSTA26
004454' 000000  006015'                    TSTA27
004455' 777777  777777                     -1

                                   ; Start test

004456' 201 00 0 00 000000'        TG23:   MOVEI   Z4                      ; get address of module start
004457' 260 17 0 00 003776*                GO      TRACE                   ; handle trace output
004460' 201 01 0 00 004605'                MOVEI   1,T23M                  ; set up microcode address
004461' 260 17 0 00 004000*                GO      TLOAD                   ; load/verify it
004462' 263 17 0 00 000000                 RTN                             ; failed - exit test
```

```


                                   ; Initialization

004463' 400 15 0 00 000000         TL23:   SETZ    ERFLG,                  ; clear error flag
004464' 260 17 0 00 004003*                GO      IPACLR                  ; clear port
004465' 402 00 0 00 004004*                SETZM   TSTSUB                  ; initialize subtest number
004466' 201 06 0 00 004500'                MOVEI   6,TS23                  ; get sstep table address

                                   ; Loop on single step table entries

004467' 260 17 0 00 004006*        TA23:   GO      BEXEC                   ; execute table entry
004470' 254 00 0 00 004477'                JRST    TX23                    ; end of sstep table
004471' 254 00 0 00 004467'                JRST    TA23                    ; keep looping after call
004472' 474 15 0 00 000000                 SETO    ERFLG,                  ; error - EBUF data incorrect

                                   ; Handle error printouts and scope looping

004473' 027 00 0 00 004602'                SCOPER  MA23                    ; print error message
004474' 254 00 0 00 004463'                JRST    TL23                    ; loop on error
004475' 254 00 0 00 004477'                JRST    TX23                    ; altmode exit
004476' 322 15 0 00 004467'                JUMPE   ERFLG,TA23              ; do next sstep table entry

                                   ; End of test

004477' 263 17 0 00 000000         TX23:   RTN                             ; return

                                   ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

004500' 300000  004522'            TS23:   ATABLE  (SSCALL,TS23A0)
004501' 101200  000011                     ATABLE  (SSSTRT,^D10,0,11,0)
004502' 000000  000000
004503' 300000  004526'                    ATABLE  (SSCALL,TS23A1)
004504' 101000  120021                     ATABLE  (SSSTRT,^D8,12,21,-2)
004505' 777777  777776
004506' 300000  004514'                    ATABLE  (SSCALL,TS23IN)
004507' 400000  004532'            TS23L:  ATABLE  (SSCHK,TS23NX)
004510' 100000  000000                     ATABLE  (SSSTRT,0,0,0,0)
004511' 000000  000000
004512' 500000  004507'                    ATABLE  (SSJRST,TS23L)
004513' 000000  000000                     ATABLE  (SSLAST)

                                   ; Initialize single step data

004514' 201 00 0 00 000001         TS23IN: MOVEI   1                       ; start with data pattern 1
004515' 202 00 0 00 010010'                MOVEM   TS23PA#                 ; save it
004516' 476 00 0 00 010011'                SETOM   TS23SG#                 ; initialize segment number
004517' 201 00 0 00 000022                 MOVEI   22                      ; get initial start address
004520' 202 00 0 00 004563'                MOVEM   TS23T2                  ;   for 1st subsection
004521' 263 17 0 00 000000                 RTN                             ; return
```

```

004522' 402 00 0 00 010005'     TS23A0: SETZM   TDATR                           ; 0+0
004523' 402 00 0 00 010007'             SETZM   TDATS
004524' 402 00 0 00 010006'             SETZM   TDATRS
004525' 263 17 0 00 000000              RTN

004526' 476 00 0 00 010005'     TS23A1: SETOM   TDATR                           ; -1+-1
004527' 476 00 0 00 010007'             SETOM   TDATS
004530' 402 00 0 00 010006'             SETZM   TDATRS
004531' 263 17 0 00 000000              RTN

                                ; Handle single step data

004532' 336 00 0 00 010010'     TS23NX: SKIPN   TS23PA                          ; done with data patterns?
004533' 263 17 0 00 000000              RTN                                     ; yes - return +1
004534' 350 01 0 00 010011'             AOS     1,TS23SG                        ; point to proper segment
004535' 303 01 0 00 000004              CAILE   1,4                             ; done with this data pattern?
                                        JRST    [MOVE   1,TS23PA                ; yes - make up next one and go
                                                 LSH    1,1                     ;   handle that one
                                                 MOVEM  1,TS23PA
                                                 SETOM  TS23SG
                                                 MOVEI  24                      ; set up subsequent start address
                                                 MOVEM  TS23T2                  ;   for 1st subsection
004536' 254 00 0 00 007232'                      JRST   TS23NX]
004537' 200 02 0 06 000000              MOVE    2,(6)                           ; get table entry
004540' 200 00 0 01 004556'             MOVE    TS23T1(1)                       ; get number of single steps
004541' 137 00 0 00 007241'             DPB     [POINT 9,2,11]                  ; insert into table entry
004542' 200 00 0 01 004563'             MOVE    TS23T2(1)                       ; get starting address
004543' 137 00 0 00 007242'             DPB     [POINT 12,2,23]                 ; insert into table entry
004544' 200 00 0 01 004570'             MOVE    TS23T3(1)                       ; get ending address
004545' 137 00 0 00 007243'             DPB     [POINT 12,2,35]                 ; insert into table entry
004546' 202 02 0 06 000000              MOVEM   2,(6)                           ; save table entry
004547' 402 00 0 00 010005'             SETZM   TDATR#                          ; clear initial data
004550' 402 00 0 00 010007'             SETZM   TDATS#                          ; clear initial data
004551' 256 00 0 01 004575'             XCT     TS23EB(1)                       ; get resulting data pattern
004552' 202 00 0 06 000001              MOVEM   1(6)                            ; save EBUF data
004553' 202 00 0 00 010006'             MOVEM   TDATRS#                         ; save result data
004554' 350 00 0 17 000000              AOS     (P)                             ; set up proper return
004555' 263 17 0 00 000000              RTN                                     ; return

                                ; Number of single steps

004556' 000000  000012          TS23T1: ^D10
004557' 000000  000011                  9
004560' 000000  000005                  5
004561' 000000  000003                  3
004562' 000000  000003                  3
```

```

                                        ; Starting address

004563' 000000  000022                  TS23T2: 22
004564' 000000  000036                          36
004565' 000000  000047                          47
004566' 000000  000062                          62
004567' 000000  000065                          65

                                        ; Ending address

004570' 000000  000035                  TS23T3: 35
004571' 000000  000046                          46
004572' 000000  000053                          53
004573' 000000  000064                          64
004574' 000000  000067                          67

                                        ; Table which calculates the result

                                        TS23EB: GO  [MOVE TS23PA          ; 0 + f1
                                                     MOVEM TDATS
004575' 260 17 0 00 007244'                          RTN]
                                                GO  [MOVE TS23PA          ; 0 + f0
                                                     SETCA 0,
                                                     MOVEM TDATS
004576' 260 17 0 00 007247'                          RTN]
                                                GO  [MOVE TS23PA          ; 1 + f0
                                                     SETCA 0,
                                                     AOS   TDATR
                                                     MOVEM TDATS
                                                     TLNE  400000
                                                     JRST  [TLZ   400000
                                                            AOS
                                                            TLO   400000
                                                            RTN]
                                                     HRLZI 400000
004577' 260 17 0 00 007257'                          RTN]
                                                GO  [MOVE TS23PA          ; f1 + f1
                                                     MOVEM TDATR
                                                     MOVEM TDATS
                                                     LSH   1
004600' 260 17 0 00 007267'                          RTN]
                                                GO  [MOVE TS23PA          ; f1 + f0
                                                     MOVEM TDATR
                                                     SETCA 0,
                                                     MOVEM TDATS
                                                     SETO  0,
004601' 260 17 0 00 007274'                          RTN]
```

```
                                 ; Error messages

004602' 140000  007302'          MA23:   MSG!TXNOT![ASCIZ /2901's not doing an R+S function properly/]
004603' 260000  006547'                  CALL!TXALL!RESPNT                ; print R+S data
004604' 000000000000#                    LAST!CALL!TXALL!SSPNT            ; print sequencer data

                                 ; Microcode:

                                 ;       Add 0 + 0, verify that this works

004605' 000000  010000           T23M:   MWORD   <ADDR=0,JMAP,J=1,SOA,AND,D=2>                   ; 0
004606' 442000  000040
004607' 000100  000000                   MWORD   <CONT,SOA,AND,D=2,B=17>                         ; 1
004610' 442007  400340
004611' 000200  000000                   MWORD   <CONT,SAB,PLUS,D=2,B=17>                        ; 2
004612' 102007  400340
004613' 000300  000000                   MWORD   <CONT,SOA,PLUS,D=2,B=16>                        ; 3
004614' 402007  000340
004615' 000400  000000                   MWORD   <CONT,SAB,OR,A=16,B=17,D=2>                     ; 4
004616' 132167  400340
004617' 000500  000000                   MWORD   <CONT,SOA,AND,D=0>                              ; 5
004620' 440000  000340
004621' 000600  000000                   MWORD   <CONT,SOQ,PLUS,D=2,B=16>                        ; 6
004622' 202007  000340
004623' 000700  000000                   MWORD   <CONT,SAB,OR,A=16,B=17,D=2>                     ; 7
004624' 132167  400340
004625' 001000  000000                   MWORD   <CONT,SOB,PLUS,D=2,B=17>                        ; 10
004626' 302007  400340
004627' 001100  112004                   MWORD   <JMAP,J=11,SOA,A=17,OR,D=1,OENA,SELE,MGC=4>     ; 11
004630' 431170  005040

                                 ;       Add -1 + -1, verify that this works

004631' 001200  130000                   MWORD   <JMAP,J=13,SOA,AND,D=2>                         ; 12
004632' 442000  000040
004633' 001300  000000                   MWORD   <CONT,SOA,XNOR,B=1,D=2>                         ; 13
004634' 472000  400340
004635' 001400  000000                   MWORD   <CONT,SOA,XNOR,B=17,D=2>                        ; 14
004636' 472007  400340
004637' 001500  000000                   MWORD   <CONT,SOA,XNOR,D=0>                             ; 15
004640' 470000  000340
004641' 001600  000000                   MWORD   <CONT,SAB,PLUS,D=2,A=1,B=17>                    ; 16
004642' 102017  400340
004643' 001700  000000                   MWORD   <CONT,SAQ,PLUS,D=2,A=1,B=16>                    ; 17
004644' 002017  000340
004645' 002000  000000                   MWORD   <CONT,SAB,AND,D=2,A=16,B=17>                    ; 20
004646' 142167  400340
004647' 002100  212004                   MWORD   <JMAP,J=21,SOA,A=17,OR,D=1,OENA,SELE,MGC=4>     ; 21
004650' 431170  005040

                                 ;       Build a data pattern 0000..001 / Left shift it once

004651' 002200  230000                   MWORD   <JMAP,J=23,SOA,AND,D=2>                         ; 22 init
004652' 442000  000040
```

```
004653' 002300  260000                  MWORD   <JMAP,J=26,SOA,PLUS,D=2,CRY>                    ; 23
004654' 402000  000440
004655' 002400  250000                  MWORD   <JMAP,J=25,SOA,OR,D=1>                          ; 24 left shift
004656' 431000  000040
004657' 002500  000000                  MWORD   <CONT,SOA,OR,D=7>                               ; 25
004660' 437000  000340

                                ;;      Add 0 + data pattern (floating 1)

004661' 002600  000000                  MWORD   <CONT,SOA,PLUS,B=1,D=2>                         ; 26
004662' 402000  400340
004663' 002700  000000                  MWORD   <CONT,SOA,PLUS,A=1,B=2,D=3>                     ; 27
004664' 403011  000340
004665' 003000  000000                  MWORD   <CONT,SOA,PLUS,A=2,B=3,D=2>                     ; 30
004666' 402021  400340
004667' 003100  000000                  MWORD   <CONT,SOB,PLUS,A=3,B=3,D=3>                     ; 31
004670' 303031  400340
004671' 003200  000000                  MWORD   <CONT,SOB,PLUS,A=3,B=3,D=2>                     ; 32
004672' 302031  400340
004673' 003300  000000                  MWORD   <CONT,SOB,PLUS,A=3,B=3,D=0>                     ; 33
004674' 300031  400340
004675' 003400  000000                  MWORD   <CONT,SOQ,PLUS,D=0>                             ; 34
004676' 200000  000340
004677' 003500  352004                  MWORD   <JMAP,J=35,SOQ,OR,D=1,OENA,SELE,MGC=4>          ; 35
004700' 231000  005040

                                ;;      Add 0 + data pattern inverted (floating 0)

004701' 003600  370000                  MWORD   <JMAP,J=37,SOA,XNOR,B=1,D=2>                    ; 36
004702' 472000  400040
004703' 003700  000000                  MWORD   <CONT,SOA,PLUS,A=1,B=2,D=2>                     ; 37
004704' 402011  000340
004705' 004000  000000                  MWORD   <CONT,SOA,PLUS,A=2,B=3,D=3>                     ; 40
004706' 403021  400340
004707' 004100  000000                  MWORD   <CONT,SOA,PLUS,A=3,B=4,D=2>                     ; 41
004710' 402032  000340
004711' 004200  000000                  MWORD   <CONT,SOB,PLUS,A=4,B=4,D=3>                     ; 42
004712' 303042  000340
004713' 004300  000000                  MWORD   <CONT,SOB,PLUS,A=4,B=4,D=2>                     ; 43
004714' 302042  000340
004715' 004400  000000                  MWORD   <CONT,SOB,PLUS,A=4,B=4,D=0>                     ; 44
004716' 300042  000340
004717' 004500  000000                  MWORD   <CONT,SOQ,PLUS,D=0>                             ; 45
004720' 200000  000340
004721' 004600  462004                  MWORD   <JMAP,J=46,SOQ,OR,D=1,OENA,SELE,MGC=4>          ; 46
004722' 231000  005040

                                ;;      Add 1 + data pattern inverted (floating 0)

004723' 004700  500000                  MWORD   <JMAP,J=50,SOA,XNOR,B=1,D=2>                    ; 47
004724' 472000  400040
004725' 005000  000000                  MWORD   <CONT,SOA,AND,D=0>                              ; 50
004726' 440000  000340
004727' 005100  000000                  MWORD   <CONT,SOQ,PLUS,D=0,CRY>                         ; 51
004730' 200000  000740
```

```
004731' 005200  000000                  MWORD   <CONT,SAQ,PLUS,A=1,B=2,D=2>                     ; 52
004732' 002011  000340
004733' 005300  532004                  MWORD   <JMAP,J=53,SOA,OR,A=2,D=1,OENA,SELE,MGC=4>      ; 53
004734' 431020  005040

                                ;;      Add f1 + f1

004735' 006200  630000                  MWORD   <ADDR=62,JMAP,J=63,SOA,OR,B=1,D=2>              ; 62
004736' 432000  400040
004737' 006300  000000                  MWORD   <CONT,SAB,PLUS,B=1,D=2>                         ; 63
004740' 102000  400340
004741' 006400  642004                  MWORD   <JMAP,J=64,SOA,OR,A=1,D=1,OENA,SELE,MGC=4>      ; 64
004742' 431010  005040

                                ;;      Add f1 + f0

004743' 006500  660000                  MWORD   <JMAP,J=66,SOA,XNOR,B=1,D=2>                    ; 65
004744' 472000  400040
004745' 006600  000000                  MWORD   <CONT,SAB,PLUS,B=1,D=2>                         ; 66
004746' 102000  400340
004747' 006700  672004                  MWORD   <JMAP,J=67,SOA,OR,A=1,D=1,OENA,SELE,MGC=4>      ; 67
004750' 431010  005040
004751' 777777  777777                  -1
```

```
                                      ;#*******************************************************************
                                      ;* Test 24 - 2901 R + S Test, With Carry
                                      ;
                                      ; Description:  Verify 0 + 0 + cry => 1
                                      ;               Verify -1 + -1 + cry => -1
                                      ;               Verify 0 + floating 1's + cry ==> floating 1's + 1
                                      ;               Verify 0 + floating 0's + cry ==> 11..11..00..00
                                      ;               Verify 1 + floating 0's + cry ==> 11..11..00..01
                                      ;               Verify that float 1's + float 1's + cry ==> 10..00..01
                                      ;               Verify that float 1's + float 0's + cry ==> 0
                                      ;
                                      ; Procedure:    Clear Port
                                      ;               Load microcode/set start address 0
                                      ;
                                      ;               Add 0 + 0, verify that this works
                                      ;               Add -1 + -1, verify that this works
                                      ;
                                      ;               Build a data pattern (floating 1)
                                      ;                 Add 0 + data pattern (floating 1)
                                      ;                 Add 0 + data pattern inverted (floating 0)
                                      ;                 Add 1 + data pattern inverted (floating 0)
                                      ;                 Add f1 + f1
                                      ;                 Add f1 + f0
                                      ;                 Verify that each works
                                      ;               Repeat with data patterns - floating 1's
                                      ;
                                      ; Failure:      ---
                                      ;#*******************************************************************

                                      ; Test data

004752' 254 00 0 00 004763'           TSTA24: JRST    TG24                    ; go start test
004753' 420402  000024                        EBUS!ALU!NDMP!ZALU!24           ; test mask
004754' 005114' 007313'                       T24M,,[ASCIZ ^2901 R + S Test - With Carry^]
004755' 006620' 000000                        [EXP MLAST!E23],,0
004756' 000000  005261'                       TSTA25                          ; failure test table
004757' 000000  005552'                       TSTA26                          ;   ...
004760' 000000  006015'                       TSTA27
004761' 000000  006306'                       TSTA30
004762' 777777  777777                        -1

                                      ; Start test

004763' 201 00 0 00 000000'           TG24:   MOVEI   Z4                      ; get address of module start
004764' 260 17 0 00 004457*                   GO      TRACE                   ; handle trace output
004765' 201 01 0 00 005114'                   MOVEI   1,T24M                  ; set up microcode address
004766' 260 17 0 00 004461*                   GO      TLOAD                   ; load/verify it
004767' 263 17 0 00 000000                    RTN                             ; failed - exit test
```

```

                                  ; Initialization

004770' 400 15 0 00 000000        TL24:   SETZ    ERFLG,                  ; clear error flag
004771' 260 17 0 00 004464*               GO      IPACLR                  ; clear port
004772' 402 00 0 00 004465*               SETZM   TSTSUB                  ; initialize subtest number
004773' 201 06 0 00 005005'               MOVEI   6,TS24                  ; get sstep table address

                                  ; Loop on single step table entries

004774' 260 17 0 00 004467*       TA24:   GO      BEXEC                   ; execute table entry
004775' 254 00 0 00 005004'               JRST    TX24                    ; end of sstep table
004776' 254 00 0 00 004774'               JRST    TA24                    ; keep looping after call
004777' 474 15 0 00 000000                SETO    ERFLG,                  ; error - EBUF data incorrect

                                  ; Handle error printouts and scope looping

005000' 027 00 0 00 005111'               SCOPER  MA24                    ; print error message
005001' 254 00 0 00 004770'               JRST    TL24                    ; loop on error
005002' 254 00 0 00 005004'               JRST    TX24                    ; altmode exit
005003' 322 15 0 00 004774'               JUMPE   ERFLG,TA24              ; do next sstep table entry

                                  ; End of test

005004' 263 17 0 00 000000        TX24:   RTN                             ; return

                                  ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

005005' 300000  005027'           TS24:   ATABLE  (SSCALL,TS24A0)
005006' 101200  000011                    ATABLE  (SSSTRT,^D10,0,11,4)
005007' 000000  000004
005010' 300000  005034'                   ATABLE  (SSCALL,TS24A1)
005011' 101000  120021                    ATABLE  (SSSTRT,^D8,12,21,-1)
005012' 777777  777777
005013' 300000  005021'                   ATABLE  (SSCALL,TS24IN)
005014' 400000  005041'           TS24L:  ATABLE  (SSCHK,TS24NX)
005015' 100000  000000                    ATABLE  (SSSTRT,0,0,0,0)
005016' 000000  000000
005017' 500000  005014'                   ATABLE  (SSJRST,TS24L)
005020' 000000  000000                    ATABLE  (SSLAST)

                                  ; Initialize single step data

005021' 201 00 0 00 000001        TS24IN: MOVEI   1                       ; start with data pattern 1
005022' 202 00 0 00 010012'               MOVEM   TS24PA#                 ; save it
005023' 476 00 0 00 010013'               SETOM   TS24SG#                 ; initialize segment number
005024' 201 00 0 00 000022                MOVEI   22                      ; get initial start address
005025' 202 00 0 00 005072'               MOVEM   TS24T2                  ;   for 1st subsection
005026' 263 17 0 00 000000                RTN                             ; return

005027' 402 00 0 00 010005'       TS24A0: SETZM   TDATR                   ; 0+0+cry
005030' 402 00 0 00 010007'               SETZM   TDATS
005031' 402 00 0 00 010006'               SETZM   TDATRS
005032' 350 00 0 00 010006'               AOS     TDATRS
005033' 263 17 0 00 000000                RTN
```

```

005034' 476 00 0 00 010005'     TS24A1: SETOM   TDATR                                   ; -1+-1+cry
005035' 476 00 0 00 010007'             SETOM   TDATS
005036' 402 00 0 00 010006'             SETZM   TDATRS
005037' 350 00 0 00 010006'             AOS     TDATRS
005040' 263 17 0 00 000000              RTN

                                ; Handle single step data

005041' 336 00 0 00 010012'     TS24NX: SKIPN   TS24PA                                  ; done with data patterns?
005042' 263 17 0 00 000000              RTN                                             ; yes - return +1
005043' 350 01 0 00 010013'             AOS     1,TS24SG                                ; point to proper segment
005044' 303 01 0 00 000004              CAILE   1,4                                     ; done with this data pattern?
                                        JRST    [MOVE   1,TS24PA                        ; yes - make up next one and go
                                                 LSH    1,1                             ;   handle that one
                                                 MOVEM  1,TS24PA
                                                 SETOM  TS24SG
                                                 MOVEI  24                              ; set up subsequent start address
                                                 MOVEM  TS24T2                          ;   for 1st subsection
005045' 254 00 0 00 007321'                      JRST   TS24NX]
005046' 200 02 0 06 000000              MOVE    2,(6)                                   ; get table entry
005047' 200 00 0 01 005065'             MOVE    TS24T1(1)                               ; get number of single steps
005050' 137 00 0 00 007241'             DPB     [POINT 9,2,11]                          ; insert into table entry
005051' 200 00 0 01 005072'             MOVE    TS24T2(1)                               ; get starting address
005052' 137 00 0 00 007242'             DPB     [POINT 12,2,23]                         ; insert into table entry
005053' 200 00 0 01 005077'             MOVE    TS24T3(1)                               ; get ending address
005054' 137 00 0 00 007243'             DPB     [POINT 12,2,35]                         ; insert into table entry
005055' 202 02 0 06 000000              MOVEM   2,(6)                                   ; save table entry
005056' 402 00 0 00 010005'             SETZM   TDATR                                   ; clear initial data
005057' 402 00 0 00 010007'             SETZM   TDATS                                   ; clear initial data
005060' 256 00 0 01 005104'             XCT     TS24EB(1)                               ; get resulting data pattern
005061' 202 00 0 06 000001              MOVEM   1(6)                                    ; save EBUF data
005062' 202 00 0 00 010006'             MOVEM   TDATRS                                  ; save result data
005063' 350 00 0 17 000000              AOS     (P)                                     ; set up proper return
005064' 263 17 0 00 000000              RTN                                             ; return

                                ; Number of single steps

005065' 000000  000012          TS24T1: ^D10
005066' 000000  000011                  9
005067' 000000  000005                  5
005070' 000000  000003                  3
005071' 000000  000003                  3

                                ; Starting address

005072' 000000  000022          TS24T2: 22
005073' 000000  000036                  36
005074' 000000  000047                  47
005075' 000000  000062                  62
005076' 000000  000065                  65
```

```

                                        ; Ending address

005077' 000000  000035                  TS24T3: 35
005100' 000000  000046                          46
005101' 000000  000053                          53
005102' 000000  000064                          64
005103' 000000  000067                          67

                                        ; Table which calculates the result

                                        TS24EB: GO      [MOVE   TS24PA          ; 0 + f1 + cry
                                                         MOVEM  TDATS
                                                         TLNN   400000
                                                         ADDI   7
                                                         TLNE   400000
                                                         TRO    7
005104' 260 17 0 00 007330'                              RTN]
                                                GO      [MOVE   1,TS24PA        ; 0 + f0 + cry
                                                         SETCA  1,
                                                         MOVEI  7
                                                         MOVEM  1,TDATS
                                                         GO     ADDLOG
005105' 260 17 0 00 007337'                              RTN]
                                                GO      [MOVE   1,TS24PA        ; 1 + f0 + cry
                                                         SETCA  1,
                                                         AOS    TDATR
                                                         MOVEM  1,TDATS
                                                         MOVEI  2
                                                         GO     ADDLOG
005106' 260 17 0 00 007345'                              RTN]
                                                GO      [MOVE   TS24PA          ; f1 + f1 + cry
                                                         MOVEM  TDATR
                                                         MOVEM  TDATS
                                                         LSH    1
                                                         TRO    1
005107' 260 17 0 00 007354'                              RTN]
                                                GO      [MOVE   TS24PA          ; f1 + f0 + cry
                                                         MOVEM  TDATR
                                                         SETCA  0,
                                                         MOVEM  TDATS
                                                         SETZ
005110' 260 17 0 00 007362'                              RTN]

                                        ; Error messages

005111' 140000  007370'                 MA24:   MSG!TXNOT![ASCIZ /2901's not doing several R+S+CRY functions properly/]
005112' 260000  006547'                         CALL!TXALL!RESPNT               ; print R+S data
005113' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```

                                ; Microcode:

                                ;       Add 0 + 0 + cry, verify that this works

005114' 000000  010000          T24M:   MWORD   <ADDR=0,JMAP,J=1,SOA,AND,D=2>                   ; 0
005115' 442000  000040
005116' 000100  000000                  MWORD   <CONT,SOA,AND,D=2,B=17>                         ; 1
005117' 442007  400340
005120' 000200  000000                  MWORD   <CONT,SAB,PLUS,CRY,D=2,B=17>                    ; 2
005121' 102007  400740
005122' 000300  000000                  MWORD   <CONT,SOA,PLUS,CRY,D=2,B=16>                    ; 3
005123' 402007  000740
005124' 000400  000000                  MWORD   <CONT,SAB,PLUS,A=16,B=17,D=2>                   ; 4
005125' 102167  400340
005126' 000500  000000                  MWORD   <CONT,SOA,AND,D=0>                              ; 5
005127' 440000  000340
005130' 000600  000000                  MWORD   <CONT,SOQ,PLUS,CRY,D=2,B=16>                    ; 6
005131' 202007  000740
005132' 000700  000000                  MWORD   <CONT,SAB,PLUS,A=16,B=17,D=2>                   ; 7
005133' 102167  400340
005134' 001000  000000                  MWORD   <CONT,SOB,PLUS,CRY,D=2,B=17>                    ; 10
005135' 302007  400740
005136' 001100  112004                  MWORD   <JMAP,J=11,SOA,A=17,OR,D=1,OENA,SELE,MGC=4>     ; 11
005137' 431170  005040

                                ;       Add -1 + -1 + cry, verify that this works

005140' 001200  130000                  MWORD   <JMAP,J=13,SOA,AND,D=2>                         ; 12
005141' 442000  000040
005142' 001300  000000                  MWORD   <CONT,SOA,XNOR,B=1,D=2>                         ; 13
005143' 472000  400340
005144' 001400  000000                  MWORD   <CONT,SOA,XNOR,B=17,D=2>                        ; 14
005145' 472007  400340
005146' 001500  000000                  MWORD   <CONT,SOA,XNOR,D=0>                             ; 15
005147' 470000  000340
005150' 001600  000000                  MWORD   <CONT,SAB,PLUS,CRY,D=2,A=1,B=17>                ; 16
005151' 102017  400740
005152' 001700  000000                  MWORD   <CONT,SAQ,PLUS,CRY,D=2,A=1,B=16>                ; 17
005153' 002017  000740
005154' 002000  000000                  MWORD   <CONT,SAB,AND,D=2,A=16,B=17>                    ; 20
005155' 142167  400340
005156' 002100  212004                  MWORD   <JMAP,J=21,SOA,A=17,OR,D=1,OENA,SELE,MGC=4>     ; 21
005157' 431170  005040

                                ;       Build a data pattern 0000..001 / Left shift it once

005160' 002200  230000                  MWORD   <JMAP,J=23,SOA,AND,D=2>                         ; 22 init
005161' 442000  000040
005162' 002300  260000                  MWORD   <JMAP,J=26,SOA,PLUS,D=2,CRY>                    ; 23
005163' 402000  000440
005164' 002400  250000                  MWORD   <JMAP,J=25,SOA,OR,D=1>                          ; 24 left shift
005165' 431000  000040
005166' 002500  000000                  MWORD   <CONT,SOA,OR,D=7>                               ; 25
005167' 437000  000340
```

```
4105
4106                                          ;;      Add 0 + data pattern (floating 1) + cry
4107
4108    005170' 002600  000000                        MWORD   <CONT,SOA,PLUS,CRY,B=1,D=2>                     ; 26
4109    005171' 402000  400740
4110    005172' 002700  000000                        MWORD   <CONT,SOA,PLUS,CRY,A=1,B=2,D=3>                 ; 27
4111    005173' 403011  000740
4112    005174' 003000  000000                        MWORD   <CONT,SOA,PLUS,CRY,A=2,B=3,D=2>                 ; 30
4113    005175' 402021  400740
4114    005176' 003100  000000                        MWORD   <CONT,SOB,PLUS,CRY,A=3,B=3,D=3>                 ; 31
4115    005177' 303031  400740
4116    005200' 003200  000000                        MWORD   <CONT,SOB,PLUS,CRY,A=3,B=3,D=2>                 ; 32
4117    005201' 302031  400740
4118    005202' 003300  000000                        MWORD   <CONT,SOB,PLUS,CRY,A=3,B=3,D=0>                 ; 33
4119    005203' 300031  400740
4120    005204' 003400  000000                        MWORD   <CONT,SOQ,PLUS,CRY,D=0>                         ; 34
4121    005205' 200000  000740
4122    005206' 003500  352004                        MWORD   <JMAP,J=35,SOQ,OR,D=1,OENA,SELE,MGC=4>          ; 35
4123    005207' 231000  005040
4124
4125                                          ;;      Add 0 + data pattern inverted (floating 0) + cry
4126
4127    005210' 003600  370000                        MWORD   <JMAP,J=37,SOA,XNOR,B=1,D=2>                    ; 36
4128    005211' 472000  400040
4129    005212' 003700  000000                        MWORD   <CONT,SOA,PLUS,CRY,A=1,B=2,D=2>                 ; 37
4130    005213' 402011  000740
4131    005214' 004000  000000                        MWORD   <CONT,SOA,PLUS,CRY,A=2,B=3,D=3>                 ; 40
4132    005215' 403021  400740
4133    005216' 004100  000000                        MWORD   <CONT,SOA,PLUS,CRY,A=3,B=4,D=2>                 ; 41
4134    005217' 402032  000740
4135    005220' 004200  000000                        MWORD   <CONT,SOB,PLUS,CRY,A=4,B=4,D=3>                 ; 42
4136    005221' 303042  000740
4137    005222' 004300  000000                        MWORD   <CONT,SOB,PLUS,CRY,A=4,B=4,D=2>                 ; 43
4138    005223' 302042  000740
4139    005224' 004400  000000                        MWORD   <CONT,SOB,PLUS,CRY,A=4,B=4,D=0>                 ; 44
4140    005225' 300042  000740
4141    005226' 004500  000000                        MWORD   <CONT,SOQ,PLUS,CRY,D=0>                         ; 45
4142    005227' 200000  000740
4143    005230' 004600  462004                        MWORD   <JMAP,J=46,SOQ,OR,D=1,OENA,SELE,MGC=4>          ; 46
4144    005231' 231000  005040
4145
4146                                          ;;      Add 1 + data pattern inverted (floating 0) + cry
4147
4148    005232' 004700  500000                        MWORD   <JMAP,J=50,SOA,XNOR,B=1,D=2>                    ; 47
4149    005233' 472000  400040
4150    005234' 005000  000000                        MWORD   <CONT,SOA,AND,D=0>                              ; 50
4151    005235' 440000  000340
4152    005236' 005100  000000                        MWORD   <CONT,SOQ,PLUS,D=0,CRY>                         ; 51
4153    005237' 200000  000740
4154    005240' 005200  000000                        MWORD   <CONT,SAQ,PLUS,CRY,A=1,B=2,D=2>                 ; 52
4155    005241' 002011  000740
4156    005242' 005300  532004                        MWORD   <JMAP,J=53,SOA,OR,A=2,D=1,OENA,SELE,MGC=4>      ; 53
4157    005243' 431020  005040
4158
4159                                          ;;      Add f1 + f1 + cry
```

```
 4160
 4161   005244' 006200  630000                  MWORD   <ADDR=62,JMAP,J=63,SOA,OR,B=1,D=2>             ; 62
 4162   005245' 432000  400040
 4163   005246' 006300  000000                  MWORD   <CONT,SAB,PLUS,CRY,B=1,D=2>                    ; 63
 4164   005247' 102000  400740
 4165   005250' 006400  642004                  MWORD   <JMAP,J=64,SOA,OR,A=1,D=1,OENA,SELE,MGC=4>     ; 64
 4166   005251' 431010  005040
 4167
 4168                                   ;;      Add f1 + f0 + cry
 4169
 4170   005252' 006500  660000                  MWORD   <JMAP,J=66,SOA,XNOR,B=1,D=2>                   ; 65
 4171   005253' 472000  400040
 4172   005254' 006600  000000                  MWORD   <CONT,SAB,PLUS,CRY,B=1,D=2>                    ; 66
 4173   005255' 102000  400740
 4174   005256' 006700  672004                  MWORD   <JMAP,J=67,SOA,OR,A=1,D=1,OENA,SELE,MGC=4>     ; 67
 4175   005257' 431010  005040
 4176   005260' 777777  777777                  -1
```

```
                                        ;#*********************************************************************
                                        ;* Test 25 - 2901 R - S Test - Without Carry
                                        ;
                                        ; Description:  Verify 0 - 0 ==> -1
                                        ;               Verify -1 - -1 ==> -1
                                        ;               Verify 0 - floating 1's ==> floating 0's
                                        ;               Verify 0 - floating 0's ==> floating 1's
                                        ;               Verify 1 - floating 0's ==> floating 1's + 1
                                        ;               Verify that float 1's - float 1's ==> -1
                                        ;               Verify that float 1's - float 0's ==> floating 1's
                                        ;
                                        ; Procedure:    Clear Port
                                        ;               Load microcode/set start address 0
                                        ;
                                        ;               Add 0 - 0, verify that this works
                                        ;               Add -1 - -1, verify that this works
                                        ;
                                        ;               Build a data pattern (floating 1)
                                        ;                 Add 0 - data pattern (floating 1)
                                        ;                 Add 0 - data pattern inverted (floating 0)
                                        ;                 Add 1 - data pattern inverted (floating 0)
                                        ;                 Add f1 - f1
                                        ;                 Add f1 - f0
                                        ;                 Verify that each works
                                        ;               Repeat with data patterns - floating 1's
                                        ;
                                        ; Failure:      ---
                                        ;#*********************************************************************

                                        ; Test data

005261' 254 00 0 00 005272'             TSTA25: JRST    TG25                    ; go start test
005262' 420402  000025                          EBUS!ALU!NDMP!ZALU!25           ; test mask
005263' 005421' 007403'                         T25M,,[ASCIZ ^2901 R - S Test - No Carry^]
005264' 006620' 000000                          [EXP MLAST!E23],,0
005265' 000000  005552'                         TSTA26                          ; failure test table
005266' 000000  006015'                         TSTA27                          ;   ...
005267' 000000  006306'                         TSTA30
005270' 000000000000#                           TSTA31
005271' 777777  777777                          -1

                                        ; Start test

005272' 201 00 0 00 000000'             TG25:   MOVEI   Z4                      ; get address of module start
005273' 260 17 0 00 004764*                     GO      TRACE                   ; handle trace output
005274' 201 01 0 00 005421'                     MOVEI   1,T25M                  ; set up microcode address
005275' 260 17 0 00 004766*                     GO      TLOAD                   ; load/verify it
005276' 263 17 0 00 000000                      RTN                             ; failed - exit test
```

```

                                ; Initialization

005277' 400 15 0 00 000000      TL25:   SETZ    ERFLG,                  ; clear error flag
005300' 260 17 0 00 004771*             GO      IPACLR                  ; clear port
005301' 402 00 0 00 004772*             SETZM   TSTSUB                  ; initialize subtest number
005302' 201 06 0 00 005314'             MOVEI   6,TS25                  ; get sstep table address

                                ; Loop on single step table entries

005303' 260 17 0 00 004774*     TA25:   GO      BEXEC                   ; execute table entry
005304' 254 00 0 00 005313'             JRST    TX25                    ; end of sstep table
005305' 254 00 0 00 005303'             JRST    TA25                    ; keep looping after call
005306' 474 15 0 00 000000              SETO    ERFLG,                  ; error - EBUF data incorrect

                                ; Handle error printouts and scope looping

005307' 027 00 0 00 005416'             SCOPER  MA25                    ; print error message
005310' 254 00 0 00 005277'             JRST    TL25                    ; loop on error
005311' 254 00 0 00 005313'             JRST    TX25                    ; altmode exit
005312' 322 15 0 00 005303'             JUMPE   ERFLG,TA25              ; do next sstep table entry

                                ; End of test

005313' 263 17 0 00 000000      TX25:   RTN                             ; return

                                ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

005314' 300000  005336'         TS25:   ATABLE  (SSCALL,TS25A0)
005315' 100700  000006                  ATABLE  (SSSTRT,7,0,6,-1)
005316' 777777  777777
005317' 300000  005342'                 ATABLE  (SSCALL,TS25A1)
005320' 100700  070015                  ATABLE  (SSSTRT,7,7,15,-1)
005321' 777777  777777
005322' 300000  005330'                 ATABLE  (SSCALL,TS25IN)
005323' 400000  005346'         TS25L:  ATABLE  (SSCHK,TS25NX)
005324' 100000  000000                  ATABLE  (SSSTRT,0,0,0,0)
005325' 000000  000000
005326' 500000  005323'                 ATABLE  (SSJRST,TS25L)
005327' 000000  000000                  ATABLE  (SSLAST)

                                ; Initialize single step data

005330' 201 00 0 00 000001      TS25IN: MOVEI   1                       ; start with data pattern 1
005331' 202 00 0 00 010014'             MOVEM   TS25PA#                 ; save it
005332' 476 00 0 00 010015'             SETOM   TS25SG#                 ; initialize segment number
005333' 201 00 0 00 000016              MOVEI   16                      ; get initial start address
005334' 202 00 0 00 005377'             MOVEM   TS25T2                  ;   for 1st subsection
005335' 263 17 0 00 000000              RTN                             ; return

005336' 402 00 0 00 010005'     TS25A0: SETZM   TDATR                   ; 0 - 0
005337' 402 00 0 00 010007'             SETZM   TDATS
005340' 476 00 0 00 010006'             SETOM   TDATRS
005341' 263 17 0 00 000000              RTN
```

```
                                       TS25A1: SETOM   TDATR                           ; -1 - -1
005342' 476 00 0 00 010005'                    SETOM   TDATS
005343' 476 00 0 00 010007'                    SETOM   TDATRS
005344' 476 00 0 00 010006'                    RTN
005345' 263 17 0 00 000000

                                       ; Handle single step data

005346' 336 00 0 00 010014'            TS25NX: SKIPN   TS25PA                          ; done with data patterns?
005347' 263 17 0 00 000000                     RTN                                     ; yes - return +1
005350' 350 01 0 00 010015'                    AOS     1,TS25SG                        ; point to proper segment
005351' 303 01 0 00 000004                     CAILE   1,4                             ; done with this data pattern?
                                               JRST    [MOVE    1,TS25PA               ; yes - make up next one and go
                                                        LSH     1,1                    ;   handle that one
                                                        MOVEM   1,TS25PA
                                                        SETOM   TS25SG
                                                        MOVEI   20                     ; set up subsequent start address
                                                        MOVEM   TS25T2                 ;   for 1st subsection
                                                        JRST    TS25NX]
005352' 254 00 0 00 007411'                    MOVE    2,(6)                           ; get table entry
005353' 200 02 0 06 000000                     MOVE    TS25T1(1)                       ; get number of single steps
005354' 200 00 0 01 005372'                    DPB     [POINT 9,2,11]                  ; insert into table entry
005355' 137 00 0 00 007241'                    MOVE    TS25T2(1)                       ; get starting address
005356' 200 00 0 01 005377'                    DPB     [POINT 12,2,23]                 ; insert into table entry
005357' 137 00 0 00 007242'                    MOVE    TS25T3(1)                       ; get ending address
005360' 200 00 0 01 005404'                    DPB     [POINT 12,2,35]                 ; insert into table entry
005361' 137 00 0 00 007243'                    MOVEM   2,(6)                           ; save table entry
005362' 202 02 0 06 000000                     SETZM   TDATR                           ; clear initial data
005363' 402 00 0 00 010005'                    SETZM   TDATS                           ; clear initial data
005364' 402 00 0 00 010007'                    XCT     TS25EB(1)                       ; get resulting data pattern
005365' 256 00 0 01 005411'                    MOVEM   1(6)                            ; save EBUF data
005366' 202 00 0 06 000001                     MOVEM   TDATRS                          ; save result data
005367' 202 00 0 00 010006'                    AOS     (P)                             ; set up proper return
005370' 350 00 0 17 000000                     RTN                                     ; return
005371' 263 17 0 00 000000

                                       ; Number of single steps

005372' 000000  000011                 TS25T1: 9
005373' 000000  000010                         8
005374' 000000  000005                         5
005375' 000000  000003                         3
005376' 000000  000003                         3

                                       ; Starting address

005377' 000000  000016                 TS25T2: 16
005400' 000000  000031                         31
005401' 000000  000041                         41
005402' 000000  000053                         53
005403' 000000  000056                         56
```

```
                                        ; Ending address

005404' 000000  000030                  TS25T3: 30
005405' 000000  000040                          40
005406' 000000  000045                          45
005407' 000000  000055                          55
005410' 000000  000060                          60

                                        ; Table which calculates the result

                                        TS25EB: GO      [MOVE   TS25PA          ; 0 - f1 ==> f0
                                                         MOVEM  TDATS
                                                         SETCA  0,
                                                         RTN]
005411' 260 17 0 00 007420'                     GO      [SETZ   0,              ; 0 - f0 ==> f1
                                                         MOVE   1,TS25PA
                                                         SETCA  1,
                                                         MOVEM  1,TDATS
                                                         MOVE   TS25PA
                                                         RTN]
005412' 260 17 0 00 007424'                     GO      [MOVEI  1               ; 1 - f0 ==> f1 + 1
                                                         AOS    TDATR
                                                         MOVE   1,TS25PA
                                                         SETCA  1,
                                                         MOVEM  1,TDATS
                                                         MOVE   TS25PA
                                                         TLNN   400000
                                                         AOS
                                                         TLNE   400000
                                                         TRO    1
                                                         RTN]
005413' 260 17 0 00 007432'                     GO      [MOVE   TS25PA          ; f1 - f1 ==> -1
                                                         MOVEM  TDATR
                                                         MOVEM  TDATS
                                                         SETO   0,
                                                         RTN]
005414' 260 17 0 00 007445'                     GO      [MOVE   1,TS25PA        ; f1 - f0 ==> f1 shifted
                                                         MOVEM  1,TDATR
                                                         SETCA  1,
                                                         MOVEM  1,TDATS
                                                         MOVE   TS25PA
                                                         LSH    1
                                                         RTN]
005415' 260 17 0 00 007452'

                                        ; Error messages

005416' 140000  007461'                 MA25:   MSG!TXNOT![ASCIZ /2901 not doing an R-S function properly/]
005417' 260000  006547'                         CALL!TXALL!RESPNT               ; print R-S data
005420' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```
                                ; Microcode:
                                ;       Do 0 - 0, verify that this works
005421' 000000  010000          T25M:   MWORD   <ADDR=0,JMAP,J=1,SOA,AND,D=2>                   ; 0
005422' 442000  000040                  MWORD   <CONT,SOA,RMIN,D=2,B=17>                        ; 1
005423' 000100  000000
005424' 422007  400340                  MWORD   <CONT,SOB,RMIN,D=2,B=17>                        ; 2
005425' 000200  000000
005426' 322007  400340                  MWORD   <CONT,SAB,RMIN,D=2,B=17>                        ; 3
005427' 000300  000000
005430' 122007  400340                  MWORD   <CONT,SOA,AND,D=0>                              ; 4
005431' 000400  000000
005432' 440000  000340                  MWORD   <CONT,SOQ,RMIN,D=2,B=16>                        ; 5
005433' 000500  000000
005434' 222007  000340                  MWORD   <JMAP,J=6,SAB,OR,A=16,B=17,D=3,OENA,SELE,MGC=4> ; 6
005435' 000600  062004
005436' 133167  405040

                                ;       Do -1 - -1, verify that this works
005437' 000700  100000                  MWORD   <JMAP,J=10,SOA,AND,D=2>                         ; 7
005440' 442000  000040                  MWORD   <CONT,SOA,XNOR,B=1,D=2>                         ; 10
005441' 001000  000000
005442' 472000  400340                  MWORD   <CONT,SOA,XNOR,B=17,D=2>                        ; 11
005443' 001100  000000
005444' 472007  400340                  MWORD   <CONT,SOA,XNOR,D=0>                             ; 12
005445' 001200  000000
005446' 470000  000340                  MWORD   <CONT,SAB,RMIN,D=2,A=1,B=17>                    ; 13
005447' 001300  000000
005450' 122017  400340                  MWORD   <CONT,SAQ,RMIN,D=2,A=1,B=16>                    ; 14
005451' 001400  000000
005452' 022017  000340                  MWORD   <JMAP,J=15,SAB,OR,D=3,A=16,B=17,OENA,SELE,MGC=4>; 15
005453' 001500  152004
005454' 133167  405040

                                ;       Build a data pattern 0000..001 / Left shift it once
005455' 001600  170000                  MWORD   <JMAP,J=17,SOA,AND,D=2>                         ; 16 init
005456' 442000  000040                  MWORD   <JMAP,J=22,SOA,PLUS,D=2,CRY>                    ; 17
005457' 001700  220000                  MWORD   <JMAP,J=21,SOA,OR,D=1>                          ; 20 left shift
005460' 402000  000440                  MWORD   <CONT,SOA,OR,D=7>                               ; 21
005461' 002000  210000
005462' 431000  000040
005463' 002100  000000
005464' 437000  000340

                                ;;      Do 0 - data pattern (floating 1)
005465' 002200  000000                  MWORD   <CONT,SOA,RMIN,B=17,D=2>                        ; 22
005466' 422007  400340                  MWORD   <CONT,SOA,OR,B=16,D=2>                          ; 23
005467' 002300  000000                  MWORD   <CONT,SOA,OR,D=0>                               ; 24
005470' 432007  000340
005471' 002400  000000
```

```
005472' 430000  000340
005473' 002500  000000                    MWORD   <CONT,SOB,RMIN,B=16,D=2>                          ; 25
005474' 322007  000340
005475' 002600  000000                    MWORD   <CONT,SAB,OR,A=16,B=17,D=2>                       ; 26
005476' 132167  400340
005477' 002700  000000                    MWORD   <CONT,SOQ,RMIN,B=16,D=2>                          ; 27
005500' 222007  000340
005501' 003000  302004                    MWORD   <JMAP,J=30,SAB,OR,A=16,B=17,D=3,OENA,SELE,MGC=4>; 30
005502' 133167  405040

                                  ;;      Do 0 - data pattern inverted (floating 0)

005503' 003100  320000                    MWORD   <JMAP,J=32,SOA,XNOR,B=1,D=2>                      ; 31
005504' 472000  400040
005505' 003200  000000                    MWORD   <CONT,SOA,XNOR,D=0>                               ; 32
005506' 470000  000340
005507' 003300  000000                    MWORD   <CONT,SOA,RMIN,A=1,B=17,D=2>                      ; 33
005510' 422017  400340
005511' 003400  000000                    MWORD   <CONT,SOA,OR,A=1,B=16,D=2>                        ; 34
005512' 432017  000340
005513' 003500  000000                    MWORD   <CONT,SOB,RMIN,B=16,D=2>                          ; 35
005514' 322007  000340
005515' 003600  000000                    MWORD   <CONT,SAB,OR,A=16,B=17,D=2>                       ; 36
005516' 132167  400340
005517' 003700  000000                    MWORD   <CONT,SOQ,RMIN,B=16,D=2>                          ; 37
005520' 222007  000340
005521' 004000  402004                    MWORD   <JMAP,J=40,SAB,OR,A=16,B=17,D=3,OENA,SELE,MGC=4>; 40
005522' 133167  405040


                                  ;;      Do 1 - data pattern inverted (floating 0)

005523' 004100  420000                    MWORD   <JMAP,J=42,SOA,XNOR,D=0>                          ; 41
005524' 470000  000040
005525' 004200  000000                    MWORD   <CONT,SOB,AND,B=1,D=2>                            ; 42
005526' 342000  400340
005527' 004300  000000                    MWORD   <CONT,SOB,PLUS,B=1,D=2,CRY>                       ; 43
005530' 302000  400740
005531' 004400  000000                    MWORD   <CONT,SAQ,RMIN,A=1,B=2,D=2>                       ; 44
005532' 022011  000340
005533' 004500  452004                    MWORD   <JMAP,J=45,SOA,OR,A=2,D=1,OENA,SELE,MGC=4>        ; 45
005534' 431020  005040


                                  ;;      Do f1 - f1

005535' 005300  540000                    MWORD   <ADDR=53,JMAP,J=54,SOA,OR,B=1,D=2>                ; 53
005536' 432000  400040
005537' 005400  000000                    MWORD   <CONT,SAB,RMIN,B=1,D=2>                           ; 54
005540' 122000  400340
005541' 005500  552004                    MWORD   <JMAP,J=55,SOA,OR,A=1,D=1,OENA,SELE,MGC=4>        ; 55
005542' 431010  005040


                                  ;;      Do f1 - f0

005543' 005600  570000                    MWORD   <JMAP,J=57,SOA,XNOR,D=0>                          ; 56
005544' 470000  000040
```

```
  4490  005545' 005700  000000                    MWORD   <CONT,SAQ,RMIN,B=1,D=2>                         ; 57
  4491  005546' 022000  400340
  4492  005547' 006000  602004                    MWORD   <JMAP,J=60,SOA,OR,A=1,D=1,OENA,SELE,MGC=4>      ; 60
  4493  005550' 431010  005040
  4494  005551' 777777  777777                    -1
```

```
                              ;#********************************************************************
                              ;* Test 26 - 2901 R - S Test - With Carry
                              ;
                              ; Description:  Verify 0 - 0 + cry ==> 0
                              ;               Verify -1 - -1 + cry ==> 0
                              ;               Verify 0 - floating 1's + cry ==> 1100...00
                              ;               Verify 0 - floating 0's + cry ==> floating 1's + 1
                              ;               Verify 1 - floating 0's + cry ==> floating 1's + 2
                              ;               Verify float 1's - float 1's + cry ==> 0
                              ;               Verify float 1's - float 0's + cry ==> floating 1's + 1
                              ;
                              ; Procedure:    Clear Port
                              ;               Load microcode/set start address 0
                              ;
                              ;               Add 0 - 0, verify that this works
                              ;               Add -1 - -1, verify that this works
                              ;
                              ;               Build a data pattern (floating 1)
                              ;                 Add 0 - data pattern (floating 1)
                              ;                 Add 0 - data pattern inverted (floating 0)
                              ;                 Add 1 - data pattern inverted (floating 0)
                              ;                 Add f1 - f1
                              ;                 Add f1 - f0
                              ;                 Verify that each works
                              ;               Repeat with data patterns - floating 1's
                              ;
                              ; Failure:      ---
                              ;#********************************************************************

                              ; Test data

005552' 254 00 0 00 005563'   TSTA26: JRST    TG26                    ; go start test
005553' 420402  000026                EBUS!ALU!NDMP!ZALU!26           ; test mask
005554' 005714' 007471'               T26M,,[ASCIZ ^2901 R - S Test - With Carry^]
005555' 006620' 000000                [EXP MLAST!E23],,0
005556' 000000  006015'               TSTA27                          ; failure test table
005557' 000000  006306'               TSTA30                          ;   ...
005560' 000000000000#                 TSTA31
005561' 000000000000#                 TSTA32
005562' 777777  777777                -1

                              ; Start test

005563' 201 00 0 00 000000'   TG26:   MOVEI   Z4                      ; get address of module start
005564' 260 17 0 00 005273*           GO      TRACE                   ; handle trace output
005565' 201 01 0 00 005714'           MOVEI   1,T26M                  ; set up microcode address
005566' 260 17 0 00 005275*           GO      TLOAD                   ; load/verify it
005567' 263 17 0 00 000000            RTN                             ; failed - exit test
```

```

                                ; Initialization

005570' 400 15 0 00 000000      TL26:   SETZ    ERFLG,                  ; clear error flag
005571' 260 17 0 00 005300*             GO      IPACLR                  ; clear port
005572' 402 00 0 00 005301*             SETZM   TSTSUB                  ; initialize subtest number
005573' 201 06 0 00 005605'             MOVEI   6,TS26                  ; get sstep table address

                                ; Loop on single step table entries

005574' 260 17 0 00 005303*     TA26:   GO      BEXEC                   ; execute table entry
005575' 254 00 0 00 005604'             JRST    TX26                    ; end of sstep table
005576' 254 00 0 00 005574'             JRST    TA26                    ; keep looping after call
005577' 474 15 0 00 000000              SETO    ERFLG,                  ; error - EBUF data incorrect

                                ; Handle error printouts and scope looping

005600' 027 00 0 00 005711'             SCOPER  MA26                    ; print error message
005601' 254 00 0 00 005570'             JRST    TL26                    ; loop on error
005602' 254 00 0 00 005604'             JRST    TX26                    ; altmode exit
005603' 322 15 0 00 005574'             JUMPE   ERFLG,TA26              ; do next sstep table entry

                                ; End of test

005604' 263 17 0 00 000000      TX26:   RTN                             ; return

                                ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

005605' 300000  005627'         TS26:   ATABLE  (SSCALL,TS26A0)
005606' 100500  000004                  ATABLE  (SSSTRT,5,0,4,0)
005607' 000000  000000
005610' 300000  005634'                 ATABLE  (SSCALL,TS26A1)
005611' 100700  050013                  ATABLE  (SSSTRT,7,5,13,0)
005612' 000000  000000
005613' 300000  005621'                 ATABLE  (SSCALL,TS26IN)
005614' 400000  005641'         TS26L:  ATABLE  (SSCHK,TS26NX)
005615' 100000  000000                  ATABLE  (SSSTRT,0,0,0,0)
005616' 000000  000000
005617' 500000  005614'                 ATABLE  (SSJRST,TS26L)
005620' 000000  000000                  ATABLE  (SSLAST)

                                ; Initialize single step data

005621' 201 00 0 00 000001      TS26IN: MOVEI   1                       ; start with data pattern 1
005622' 202 00 0 00 010016'             MOVEM   TS26PA#                 ; save it
005623' 476 00 0 00 010017'             SETOM   TS26SG#                 ; initialize segment number
005624' 201 00 0 00 000016              MOVEI   16                      ; get initial start address
005625' 202 00 0 00 005672'             MOVEM   TS26T2                  ;   for 1st subsection
005626' 263 17 0 00 000000              RTN                             ; return

005627' 402 00 0 00 010005'     TS26A0: SETZM   TDATR                   ; 0 - 0 + cry
005630' 402 00 0 00 010007'             SETZM   TDATS
005631' 402 00 0 00 010006'             SETZM   TDATRS
005632' 350 00 0 00 010006'             AOS     TDATRS
005633' 263 17 0 00 000000              RTN
```

```

005634' 476 00 0 00 010005'      TS26A1: SETOM   TDATR                          ; -1 - -1 + cry
005635' 476 00 0 00 010007'              SETOM   TDATS
005636' 402 00 0 00 010006'              SETZM   TDATRS
005637' 350 00 0 00 010006'              AOS     TDATRS
005640' 263 17 0 00 000000               RTN

                                 ; Handle single step data

005641' 336 00 0 00 010016'      TS26NX: SKIPN   TS26PA                         ; done with data patterns?
005642' 263 17 0 00 000000               RTN                                    ; yes - return +1
005643' 350 01 0 00 010017'              AOS     1,TS26SG                       ; point to proper segment
005644' 303 01 0 00 000004               CAILE   1,4                            ; done with this data pattern?
                                         JRST    [MOVE   1,TS26PA               ; yes - make up next one and go
                                                  LSH    1,1                    ;   handle that one
                                                  MOVEM  1,TS26PA
                                                  SETOM  TS26SG
                                                  MOVEI  20                     ; set up subsequent start address
                                                  MOVEM  TS26T2                 ;   for 1st subsection
                                                  JRST   TS26NX]
005645' 254 00 0 00 007477'
005646' 200 02 0 06 000000               MOVE    2,(6)                          ; get table entry
005647' 200 00 0 01 005665'              MOVE    TS26T1(1)                      ; get number of single steps
005650' 137 00 0 00 007241'              DPB     [POINT 9,2,11]                 ; insert into table entry
005651' 200 00 0 01 005672'              MOVE    TS26T2(1)                      ; get starting address
005652' 137 00 0 00 007242'              DPB     [POINT 12,2,23]                ; insert into table entry
005653' 200 00 0 01 005677'              MOVE    TS26T3(1)                      ; get ending address
005654' 137 00 0 00 007243'              DPB     [POINT 12,2,35]                ; insert into table entry
005655' 202 02 0 06 000000               MOVEM   2,(6)                          ; save table entry
005656' 402 00 0 00 010005'              SETZM   TDATR                          ; clear initial data
005657' 402 00 0 00 010007'              SETZM   TDATS                          ; clear initial data
005660' 256 00 0 01 005704'              XCT     TS26EB(1)                      ; get resulting data pattern
005661' 202 00 0 06 000001               MOVEM   1(6)                           ; save EBUF data
005662' 202 00 0 00 010006'              MOVEM   TDATRS                         ; save result data
005663' 350 00 0 17 000000               AOS     (P)                            ; set up proper return
005664' 263 17 0 00 000000               RTN                                    ; return

                                 ; Number of single steps

005665' 000000  000004           TS26T1: 4
005666' 000000  000004                   4
005667' 000000  000005                   5
005670' 000000  000003                   3
005671' 000000  000003                   3

                                 ; Starting address

005672' 000000  000016           TS26T2: 16
005673' 000000  000024                   24
005674' 000000  000041                   41
005675' 000000  000053                   53
005676' 000000  000056                   56
```

```

                                ; Ending address

005677' 000000  000023          TS26T3: 23
005700' 000000  000026                  26
005701' 000000  000045                  45
005702' 000000  000055                  55
005703' 000000  000060                  60

                                ; Table which calculates the result

                                TS26EB: GO      [MOVE   TS26PA          ; 0 - f1 + cry ==> 1100..00
                                                 MOVEM  TDATS
                                                 SETCA
                                                 TLNE   400000
                                                 JRST   [TLZ    400000
                                                         AOS
                                                         TLO    400000
                                                         RTN]
                                                 HRLZI  400000
005704' 260 17 0 00 007506'                      RTN]
                                        GO      [MOVE   TS26PA          ; 0 - f0 + cry ==> f1 + 1
                                                 SETCA
                                                 MOVEM  TDATS
                                                 MOVE   TS26PA
                                                 TLNN   400000
                                                 AOS
                                                 TLNE   400000
                                                 TRO    1
005705' 260 17 0 00 007515'                      RTN]
                                        GO      [MOVE   TS26PA          ; 1 - f0 + cry ==> f1 + 2
                                                 SETCA
                                                 MOVEM  TDATS
                                                 AOS    TDATR
                                                 MOVE   TS26PA
                                                 TLNN   400000
                                                 ADDI   2
                                                 TLNE   400000
                                                 TRO    2
005706' 260 17 0 00 007526'                      RTN]
                                        GO      [MOVE   TS26PA          ; f1 - f1 + cry ==> 0
                                                 MOVEM  TDATR
                                                 MOVEM  TDATS
                                                 SETZ
005707' 260 17 0 00 007540'                      RTN]
```

```
                                    GO      [MOVE   TS26PA          ; f1 - f0 + cry ==> f1 shifted + 1
                                             MOVEM  TDATR
                                             SETCA
                                             MOVEM  TDATS
                                             MOVE   TS26PA
                                             LSH    1
                                             TLNN   400000
                                             AOS
                                             TLNE   400000
                                             TRO    1
005710' 260 17 0 00 007545'                  RTN]

                              ; Error messages

005711' 140000  007560'       MA26:   MSG!TXNOT![ASCIZ /2901 not doing an R-S+CRY function properly/]
005712' 260000  006547'               CALL!TXALL!RESPNT               ; print R-S data
005713' 000000000000#                 LAST!CALL!TXALL!SSPNT           ; print sequencer data

                              ; Microcode:

                              ;       Do 0 - 0 + cry, verify that this works

005714' 000000  010000        T26M:   MWORD   <ADDR=0,JMAP,J=1,SOA,AND,D=2>                       ; 0
005715' 442000  000040
005716' 000100  000000                MWORD   <CONT,SOA,RMIN,CRY,D=2,B=17>                        ; 1
005717' 422007  400740
005720' 000200  000000                MWORD   <CONT,SOA,AND,D=0>                                  ; 2
005721' 440000  000340
005722' 000300  000000                MWORD   <CONT,SOQ,RMIN,CRY,D=2,B=16>                        ; 3
005723' 222007  000740
005724' 000400  042004                MWORD   <JMAP,J=4,SAB,PLUS,A=16,B=17,D=1,OENA,SELE,MGC=4> ; 4
005725' 101167  405040

                              ;       Do -1 - -1 + cry, verify that this works

005726' 000500  060000                MWORD   <JMAP,J=6,SOA,AND,D=2>                              ; 5
005727' 442000  000040
005730' 000600  000000                MWORD   <CONT,SOA,XNOR,B=1,D=2>                             ; 6
005731' 472000  400340
005732' 000700  000000                MWORD   <CONT,SOA,XNOR,B=17,D=2>                            ; 7
005733' 472007  400340
005734' 001000  000000                MWORD   <CONT,SOA,XNOR,D=0>                                 ; 10
005735' 470000  000340
005736' 001100  000000                MWORD   <CONT,SAB,RMIN,CRY,D=2,A=1,B=17>                    ; 11
005737' 122017  400740
005740' 001200  000000                MWORD   <CONT,SAQ,RMIN,CRY,D=2,A=1,B=16>                    ; 12
005741' 022017  000740
005742' 001300  132004                MWORD   <JMAP,J=13,SAB,PLUS,D=1,A=16,B=17,OENA,SELE,MGC=4> ; 13
005743' 101167  405040

                              ;       Build a data pattern 0000..001 / Left shift it once

005744' 001600  170000                MWORD   <ADDR=16,JMAP,J=17,SOA,AND,D=2>                     ; 16 init
005745' 442000  000040
005746' 001700  220000                MWORD   <JMAP,J=22,SOA,PLUS,D=2,CRY>                        ; 17
```

```
 4750   005747' 402000  000440                  MWORD   <JMAP,J=21,SOA,OR,D=1>                            ; 20 left shift
 4751   005750' 002000  210000
 4752   005751' 431000  000040                  MWORD   <CONT,SOA,OR,D=7>                                 ; 21
 4753   005752' 002100  000000
 4754   005753' 437000  000340
 4755
 4756                                   ;;      Do 0 - data pattern (floating 1) + cry
 4757
 4758   005754' 002200  000000                  MWORD   <CONT,SOA,RMIN,CRY,B=17,D=2>                      ; 22
 4759   005755' 422007  400740
 4760   005756' 002300  232004                  MWORD   <JMAP,J=23,SOA,OR,A=17,D=1,OENA,SELE,MGC=4>       ; 23
 4761   005757' 431170  005040
 4762
 4763                                   ;;      Do 0 - data pattern inverted (floating 0) + cry
 4764
 4765   005760' 002400  250000                  MWORD   <JMAP,J=25,SOA,XNOR,B=1,D=2>                      ; 24
 4766   005761' 472000  400040
 4767   005762' 002500  000000                  MWORD   <CONT,SOA,RMIN,CRY,A=1,B=17,D=2>                  ; 25
 4768   005763' 422017  400740
 4769   005764' 002600  262004                  MWORD   <JMAP,J=26,SOA,OR,A=17,D=1,OENA,SELE,MGC=4>       ; 26
 4770   005765' 431170  005040
 4771
 4772                                   ;;      Do 1 - data pattern inverted (floating 0) + cry
 4773
 4774   005766' 004100  420000                  MWORD   <ADDR=41,JMAP,J=42,SOA,XNOR,D=0>                  ; 41
 4775   005767' 470000  000040
 4776   005770' 004200  000000                  MWORD   <CONT,SOB,AND,B=1,D=2>                            ; 42
 4777   005771' 342000  400340
 4778   005772' 004300  000000                  MWORD   <CONT,SOB,PLUS,B=1,D=2,CRY>                       ; 43
 4779   005773' 302000  400740
 4780   005774' 004400  000000                  MWORD   <CONT,SAQ,RMIN,CRY,A=1,B=2,D=2>                   ; 44
 4781   005775' 022011  000740
 4782   005776' 004500  452004                  MWORD   <JMAP,J=45,SOA,OR,A=2,D=1,OENA,SELE,MGC=4>        ; 45
 4783   005777' 431020  005040
 4784
 4785                                   ;;      Do f1 - f1 + cry
 4786
 4787   006000' 005300  540000                  MWORD   <ADDR=53,JMAP,J=54,SOA,OR,B=1,D=2>                ; 53
 4788   006001' 432000  400040
 4789   006002' 005400  000000                  MWORD   <CONT,SAB,RMIN,CRY,B=1,D=2>                       ; 54
 4790   006003' 122000  400740
 4791   006004' 005500  552004                  MWORD   <JMAP,J=55,SOA,OR,A=1,D=1,OENA,SELE,MGC=4>        ; 55
 4792   006005' 431010  005040
 4793
 4794                                   ;;      Do f1 - f0 + cry
 4795
 4796   006006' 005600  570000                  MWORD   <JMAP,J=57,SOA,XNOR,D=0>                          ; 56
 4797   006007' 470000  000040
 4798   006010' 005700  000000                  MWORD   <CONT,SAQ,RMIN,CRY,B=1,D=2>                       ; 57
 4799   006011' 022000  400740
 4800   006012' 006000  602004                  MWORD   <JMAP,J=60,SOA,OR,A=1,D=1,OENA,SELE,MGC=4>        ; 60
 4801   006013' 431010  005040
 4802   006014' 777777  777777                  -1
```

```
                                          ;#*****************************************************************
                                          ;* Test 27 - 2901 S - R Test - Without Carry
                                          ;
                                          ; Description:  Verify 0 - 0 ==> -1
                                          ;               Verify -1 - -1 ==> -1
                                          ;               Verify floating 1's - 0 ==> 00..0077..77
                                          ;               Verify floating 0's - 0 ==> 77..7377..76
                                          ;               Verify floating 0's - 1 ==> 77..7377..75
                                          ;               Verify that float 1's - float 1's ==> -1
                                          ;               Verify that float 0's - float 1's ==> -2
                                          ;
                                          ; Procedure:    Clear Port
                                          ;               Load microcode/set start address 0
                                          ;
                                          ;               Add 0 - 0, verify that this works
                                          ;               Add -1 - -1, verify that this works
                                          ;
                                          ;               Build a data pattern (floating 1)
                                          ;                 Do data pattern (floating 1) - 0
                                          ;                 Do data pattern inverted (floating 0) - 0
                                          ;                 Do data pattern inverted (floating 0) - 1
                                          ;                 Do f1 - f1
                                          ;                 Do f0 - f1
                                          ;                 Verify that each works
                                          ;               Repeat with data patterns - floating 1's
                                          ;
                                          ; Failure:      ---
                                          ;#*****************************************************************

                                          ; Test data

 006015' 254 00 0 00 006026'              TSTA27: JRST    TG27                    ; go start test
 006016' 420402  000027                          EBUS!ALU!NDMP!ZALU!27            ; test mask
 006017' 006155' 007571'                         T27M,,[ASCIZ ^2901 S - R Test - No Carry^]
 006020' 006620' 000000                          [EXP MLAST!E23],,0
 006021' 000000  006306'                         TSTA30                          ; failure test table
 006022' 000000000000#                           TSTA31                          ;   ...
 006023' 000000000000#                           TSTA32
 006024' 000000000000#                           TSTA33
 006025' 777777  777777                          -1

                                          ; Start test

 006026' 201 00 0 00 000000'              TG27:   MOVEI   Z4                      ; get address of module start
 006027' 260 17 0 00 005564*                     GO      TRACE                   ; handle trace output
 006030' 201 01 0 00 006155'                     MOVEI   1,T27M                  ; set up microcode address
 006031' 260 17 0 00 005566*                     GO      TLOAD                   ; load/verify it
 006032' 263 17 0 00 000000                      RTN                             ; failed - exit test
```

```

                                  ; Initialization

006033' 400 15 0 00 000000        TL27:   SETZ    ERFLG,              ; clear error flag
006034' 260 17 0 00 005571*               GO      IPACLR              ; clear port
006035' 402 00 0 00 005572*               SETZM   TSTSUB              ; initialize subtest number
006036' 201 06 0 00 006050'               MOVEI   6,TS27              ; get sstep table address

                                  ; Loop on single step table entries

006037' 260 17 0 00 005574*       TA27:   GO      BEXEC               ; execute table entry
006040' 254 00 0 00 006047'               JRST    TX27                ; end of sstep table
006041' 254 00 0 00 006037'               JRST    TA27                ; keep looping after call
006042' 474 15 0 00 000000                SETO    ERFLG,              ; error - EBUF data incorrect

                                  ; Handle error printouts and scope looping

006043' 027 00 0 00 006152'               SCOPER  MA27                ; print error message
006044' 254 00 0 00 006033'               JRST    TL27                ; loop on error
006045' 254 00 0 00 006047'               JRST    TX27                ; altmode exit
006046' 322 15 0 00 006037'               JUMPE   ERFLG,TA27          ; do next sstep table entry

                                  ; End of test

006047' 263 17 0 00 000000        TX27:   RTN                         ; return

                                  ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

006050' 300000  006072'           TS27:   ATABLE  (SSCALL,TS27A0)
006051' 100700  000006                    ATABLE  (SSSTRT,7,0,6,-1)
006052' 777777  777777
006053' 300000  006076'                   ATABLE  (SSCALL,TS27A1)
006054' 100700  070015                    ATABLE  (SSSTRT,7,7,15,-1)
006055' 777777  777777
006056' 300000  006064'                   ATABLE  (SSCALL,TS27IN)
006057' 400000  006102'           TS27L:  ATABLE  (SSCHK,TS27NX)
006060' 100000  000000                    ATABLE  (SSSTRT,0,0,0,0)
006061' 000000  000000
006062' 500000  006057'                   ATABLE  (SSJRST,TS27L)
006063' 000000  000000                    ATABLE  (SSLAST)

                                  ; Initialize single step data

006064' 201 00 0 00 000001        TS27IN: MOVEI   1                   ; start with data pattern 1
006065' 202 00 0 00 010020'               MOVEM   TS27PA#             ; save it
006066' 476 00 0 00 010021'               SETOM   TS27SG#             ; initialize segment number
006067' 201 00 0 00 000016                MOVEI   16                  ; get initial start address
006070' 202 00 0 00 006133'               MOVEM   TS27T2              ;   for 1st subsection
006071' 263 17 0 00 000000                RTN                         ; return

006072' 402 00 0 00 010005'       TS27A0: SETZM   TDATR               ; 0 - 0
006073' 402 00 0 00 010007'               SETZM   TDATS
006074' 402 00 0 00 010006'               SETZM   TDATRS
006075' 263 17 0 00 000000                RTN
```

```

006076' 476 00 0 00 010005'     TS27A1: SETOM   TDATR                           ; -1 - -1
006077' 476 00 0 00 010007'             SETOM   TDATS
006100' 402 00 0 00 010006'             SETZM   TDATRS
006101' 263 17 0 00 000000              RTN

                                ; Handle single step data

006102' 336 00 0 00 010020'     TS27NX: SKIPN   TS27PA                          ; done with data patterns?
006103' 263 17 0 00 000000              RTN                                     ; yes - return +1
006104' 350 01 0 00 010021'             AOS     1,TS27SG                        ; point to proper segment
006105' 303 01 0 00 000004              CAILE   1,4                             ; done with this data pattern?
                                        JRST    [MOVE   1,TS27PA                ; yes - make up next one and go
                                                 LSH    1,1                     ;   handle that one
                                                 MOVEM  1,TS27PA
                                                 SETOM  TS27SG
                                                 MOVEI  20                      ; set up subsequent start address
                                                 MOVEM  TS27T2                  ;   for 1st subsection
                                                 JRST   TS27NX]
006106' 254 00 0 00 007577'
006107' 200 02 0 06 000000              MOVE    2,(6)                           ; get table entry
006110' 200 00 0 01 006126'             MOVE    TS27T1(1)                       ; get number of single steps
006111' 137 00 0 00 007241'             DPB     [POINT 9,2,11]                  ; insert into table entry
006112' 200 00 0 01 006133'             MOVE    TS27T2(1)                       ; get starting address
006113' 137 00 0 00 007242'             DPB     [POINT 12,2,23]                 ; insert into table entry
006114' 200 00 0 01 006140'             MOVE    TS27T3(1)                       ; get ending address
006115' 137 00 0 00 007243'             DPB     [POINT 12,2,35]                 ; insert into table entry
006116' 202 02 0 06 000000              MOVEM   2,(6)                           ; save table entry
006117' 402 00 0 00 010005'             SETZM   TDATR                           ; clear initial data
006120' 402 00 0 00 010007'             SETZM   TDATS                           ; clear initial data
006121' 256 00 0 01 006145'             XCT     TS27EB(1)                       ; get resulting data pattern
006122' 202 00 0 06 000001              MOVEM   1(6)                            ; save EBUF data
006123' 202 00 0 00 010006'             MOVEM   TDATRS                          ; save result data
006124' 350 00 0 17 000000              AOS     (P)                             ; set up proper return
006125' 263 17 0 00 000000              RTN                                     ; return

                                ; Number of single steps

006126' 000000  000011          TS27T1: 9
006127' 000000  000010                  8
006130' 000000  000005                  5
006131' 000000  000003                  3
006132' 000000  000003                  3

                                ; Starting address

006133' 000000  000016          TS27T2: 16
006134' 000000  000031                  31
006135' 000000  000041                  41
006136' 000000  000053                  53
006137' 000000  000056                  56
```

```

                                        ; Ending address

006140' 000000  000030                  TS27T3: 30
006141' 000000  000040                          40
006142' 000000  000045                          45
006143' 000000  000055                          55
006144' 000000  000060                          60

                                        ; Table which calculates the result

                                        TS27EB: GO      [MOVE   1,TS27PA        ; f1 - 0 ==> 00..0077..77
                                                         MOVEM  1,TDATS
                                                         SETO   0,
                                                         GO     ADDLOG
006145' 260 17 0 00 007606'                              RTN]
                                                GO      [SETO   0,              ; f0 - 0 ==> 77..7377..76
                                                         MOVE   1,TS27PA
                                                         SETCA  1,
                                                         MOVEM  1,TDATS
                                                         GO     ADDLOG
006146' 260 17 0 00 007613'                              RTN]
                                                GO      [MOVEI  1               ; f0 - 1 ==> 77..7377..75
                                                         MOVEM  TDATR
                                                         MOVE   1,TS27PA
                                                         SETCA  1,
                                                         MOVEM  1,TDATS
                                                         MOVE   [-2]
                                                         GO     ADDLOG
006147' 260 17 0 00 007622'                              RTN]
                                                GO      [MOVE   TS27PA          ; f1 - f1 ==> -1
                                                         MOVEM  TDATR
                                                         MOVEM  TDATS
                                                         SETO
006150' 260 17 0 00 007632'                              RTN]
                                                GO      [MOVE   1,TS27PA        ; f0 - f1 ==> -2
                                                         MOVEM  1,TDATR
                                                         SETCA  1,
                                                         MOVEM  1,TDATS
                                                         MOVE   1
                                                         GO     ADDLOG
006151' 260 17 0 00 007637'                              RTN]

                                        ; Error messages

006152' 140000  007646'                 MA27:   MSG!TXNOT![ASCIZ /2901 not doing an S-R function properly/]
006153' 260000  006547'                         CALL!TXALL!RESPNT               ; print S-R data
006154' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```

                                ; Microcode:

                                ;       Do 0 - 0, verify that this works

006155' 000000  010000          T27M:   MWORD   <ADDR=0,JMAP,J=1,SOA,AND,D=2>                   ; 0
006156' 442000  000040
006157' 000100  000000                  MWORD   <CONT,SOA,SMIN,D=2,B=17>                        ; 1
006160' 412007  400340
006161' 000200  000000                  MWORD   <CONT,SOB,SMIN,D=2,B=17>                        ; 2
006162' 312007  400340
006163' 000300  000000                  MWORD   <CONT,SAB,SMIN,D=2,B=17>                        ; 3
006164' 112007  400340
006165' 000400  000000                  MWORD   <CONT,SOA,AND,D=0>                              ; 4
006166' 440000  000340
006167' 000500  000000                  MWORD   <CONT,SOQ,SMIN,D=2,B=16>                        ; 5
006170' 212007  000340
006171' 000600  062004                  MWORD   <JMAP,J=6,SAB,OR,A=16,B=17,D=1,OENA,SELE,MGC=4> ; 6
006172' 131167  405040

                                ;       Do -1 - -1, verify that this works

006173' 000700  100000                  MWORD   <JMAP,J=10,SOA,AND,D=2>                         ; 7
006174' 442000  000040
006175' 001000  000000                  MWORD   <CONT,SOA,XNOR,B=1,D=2>                         ; 10
006176' 472000  400340
006177' 001100  000000                  MWORD   <CONT,SOA,XNOR,B=17,D=2>                        ; 11
006200' 472007  400340
006201' 001200  000000                  MWORD   <CONT,SOA,XNOR,D=0>                             ; 12
006202' 470000  000340
006203' 001300  000000                  MWORD   <CONT,SAB,SMIN,D=2,A=1,B=17>                    ; 13
006204' 112017  400340
006205' 001400  000000                  MWORD   <CONT,SAQ,SMIN,D=2,A=1,B=16>                    ; 14
006206' 012017  000340
006207' 001500  152004                  MWORD   <JMAP,J=15,SAB,OR,D=1,A=16,B=17,OENA,SELE,MGC=4>; 15
006210' 131167  405040

                                ;       Build a data pattern 0000..001 / Left shift it once

006211' 001600  170000                  MWORD   <JMAP,J=17,SOA,AND,D=2>                         ; 16 init
006212' 442000  000040
006213' 001700  220000                  MWORD   <JMAP,J=22,SOA,PLUS,D=2,CRY>                    ; 17
006214' 402000  000440
006215' 002000  210000                  MWORD   <JMAP,J=21,SOA,OR,D=1>                          ; 20 left shift
006216' 431000  000040
006217' 002100  000000                  MWORD   <CONT,SOA,OR,D=7>                               ; 21
006220' 437000  000340

                                ;;      Do data pattern (floating 1) - 0

006221' 002200  000000                  MWORD   <CONT,SOA,SMIN,B=17,D=2>                        ; 22
006222' 412007  400340
006223' 002300  000000                  MWORD   <CONT,SOA,OR,B=16,D=2>                          ; 23
006224' 432007  000340
006225' 002400  000000                  MWORD   <CONT,SOA,OR,D=0>                               ; 24
```

```
006226' 430000  000340          MWORD   <CONT,SOB,SMIN,B=16,D=2>                        ; 25
006227' 002500  000000
006230' 312007  000340          MWORD   <CONT,SAB,OR,A=16,B=17,D=2>                     ; 26
006231' 002600  000000
006232' 132167  400340          MWORD   <CONT,SOQ,SMIN,B=16,D=2>                        ; 27
006233' 002700  000000
006234' 212007  000340          MWORD   <JMAP,J=30,SAB,OR,A=16,B=17,D=1,OENA,SELE,MGC=4>; 30
006235' 003000  302004
006236' 131167  405040

                        ;;      Do data pattern inverted (floating 0) - 0

006237' 003100  320000          MWORD   <JMAP,J=32,SOA,XNOR,B=1,D=2>                    ; 31
006240' 472000  400040
006241' 003200  000000          MWORD   <CONT,SOA,XNOR,D=0>                             ; 32
006242' 470000  000340
006243' 003300  000000          MWORD   <CONT,SOA,A=1,SMIN,B=17,D=2>                    ; 33
006244' 412017  400340
006245' 003400  000000          MWORD   <CONT,SOA,A=1,OR,B=16,D=2>                      ; 34
006246' 432017  000340
006247' 003500  000000          MWORD   <CONT,SOB,SMIN,B=16,D=2>                        ; 35
006250' 312007  000340
006251' 003600  000000          MWORD   <CONT,SAB,OR,A=16,B=17,D=2>                     ; 36
006252' 132167  400340
006253' 003700  000000          MWORD   <CONT,SOQ,SMIN,B=16,D=2>                        ; 37
006254' 212007  000340
006255' 004000  402004          MWORD   <JMAP,J=40,SAB,OR,A=16,B=17,D=1,OENA,SELE,MGC=4>; 40
006256' 131167  405040

                        ;;      Do data pattern inverted (floating 0) - 1

006257' 004100  420000          MWORD   <JMAP,J=42,SOA,XNOR,D=0>                        ; 41
006260' 470000  000040
006261' 004200  000000          MWORD   <CONT,SOB,AND,B=1,D=2>                          ; 42
006262' 342000  400340
006263' 004300  000000          MWORD   <CONT,SOB,PLUS,B=1,D=2,CRY>                     ; 43
006264' 302000  400740
006265' 004400  000000          MWORD   <CONT,SAQ,SMIN,A=1,B=2,D=2>                     ; 44
006266' 012011  000340
006267' 004500  452004          MWORD   <JMAP,J=45,SOA,OR,A=2,D=1,OENA,SELE,MGC=4>      ; 45
006270' 431020  005040

                        ;;      Do f1 - f1

006271' 005300  540000          MWORD   <ADDR=53,JMAP,J=54,SOA,OR,B=1,D=2>              ; 53
006272' 432000  400040
006273' 005400  000000          MWORD   <CONT,SAB,SMIN,B=1,D=2>                         ; 54
006274' 112000  400340
006275' 005500  552004          MWORD   <JMAP,J=55,SOA,OR,A=1,D=1,OENA,SELE,MGC=4>      ; 55
006276' 431010  005040

                        ;;      Do f0 - f1

006277' 005600  570000          MWORD   <JMAP,J=57,SOA,XNOR,D=0>                        ; 56
006300' 470000  000040
```

```
DFPTA4  MACRO %53B(1242) 08:07  1-Jul-85 Page 64-2
DFPTA4  MAC      18-Apr-85 10:59          2901 ALU Function Tests

  5114  006301' 005700  000000                          MWORD   <CONT,SAQ,SMIN,B=1,D=2>                         ; 57
  5115  006302' 012000  400340
  5116  006303' 006000  602004                          MWORD   <JMAP,J=60,SOA,OR,A=1,D=1,OENA,SELE,MGC=4>      ; 60
  5117  006304' 431010  005040
  5118  006305' 777777  777777                          -1
```

```
                                     ;#*******************************************************************
                                     ;* Test 30 - 2901 S - R Test - With Carry
                                     ;
                                     ; Description:  Verify 0 - 0 + cry ==> 0
                                     ;               Verify -1 - -1 + cry ==> 0
                                     ;               Verify floating 1's - 0 + cry ==> 00..0100..00
                                     ;               Verify floating 0's - 0 + cry ==> 77..7377..77
                                     ;               Verify floating 0's - 1 + cry ==> 77..7377..76
                                     ;               Verify that float 1's - float 1's + cry ==> 0
                                     ;               Verify that float 0's - float 1's + cry ==> -1
                                     ;
                                     ; Procedure:    Clear Port
                                     ;               Load microcode/set start address 0
                                     ;
                                     ;               Add 0 - 0, verify that this works
                                     ;               Add -1 - -1, verify that this works
                                     ;
                                     ;               Build a data pattern (floating 1)
                                     ;                 Do data pattern (floating 1) - 0
                                     ;                 Do data pattern inverted (floating 0) - 0
                                     ;                 Do data pattern inverted (floating 0) - 1
                                     ;                 Do f1 - f1
                                     ;                 Do f0 - f1
                                     ;                 Verify that each works
                                     ;               Repeat with data patterns - floating 1's
                                     ;
                                     ; Failure:      ---
                                     ;#*******************************************************************

                                     ; Test data

006306' 254 00 0 00 006317'          TSTA30: JRST    TG30                          ; go start test
006307' 420402  000030                       EBUS!ALU!NDMP!ZALU!30                 ; test mask
006310' 006446' 007656'                      T30M,,[ASCIZ ^2901 S - R Test - With Carry^]
006311' 006620' 000000                       [EXP MLAST!E23],,0
006312' 000000000000#                        TSTA31                                ; failure test table
006313' 000000000000#                        TSTA32                                ;    ...
006314' 000000000000#                        TSTA33
006315' 000000000000#                        TSTA34
006316' 777777  777777                       -1

                                     ; Start test

006317' 201 00 0 00 000000'          TG30:   MOVEI   Z4                            ; get address of module start
006320' 260 17 0 00 006027*                  GO      TRACE                         ; handle trace output
006321' 201 01 0 00 006446'                  MOVEI   1,T30M                        ; set up microcode address
006322' 260 17 0 00 006031*                  GO      TLOAD                         ; load/verify it
006323' 263 17 0 00 000000                   RTN                                   ; failed - exit test
```

```

                                    ; Initialization

006324' 400 15 0 00 000000          TL30:   SETZ    ERFLG,                  ; clear error flag
006325' 260 17 0 00 006034*                 GO      IPACLR                  ; clear port
006326' 402 00 0 00 006035*                 SETZM   TSTSUB                  ; initialize subtest number
006327' 201 06 0 00 006341'                 MOVEI   6,TS30                  ; get sstep table address

                                    ; Loop on single step table entries

006330' 260 17 0 00 006037*         TA30:   GO      BEXEC                   ; execute table entry
006331' 254 00 0 00 006340'                 JRST    TX30                    ; end of sstep table
006332' 254 00 0 00 006330'                 JRST    TA30                    ; keep looping after call
006333' 474 15 0 00 000000                  SETO    ERFLG,                  ; error - EBUF data incorrect

                                    ; Handle error printouts and scope looping

006334' 027 00 0 00 006443'                 SCOPER  MA30                    ; print error message
006335' 254 00 0 00 006324'                 JRST    TL30                    ; loop on error
006336' 254 00 0 00 006340'                 JRST    TX30                    ; altmode exit
006337' 322 15 0 00 006330'                 JUMPE   ERFLG,TA30              ; do next sstep table entry

                                    ; End of test

006340' 263 17 0 00 000000          TX30:   RTN                             ; return

                                    ; Sstep table, with EBUF data, as:  (CMD,#Steps,Start addr,End Addr,EBUF data)

006341' 300000  006363'             TS30:   ATABLE  (SSCALL,TS30A0)
006342' 100500  000004                      ATABLE  (SSSTRT,5,0,4,0)
006343' 000000  000000
006344' 300000  006367'                     ATABLE  (SSCALL,TS30A1)
006345' 100700  050013                      ATABLE  (SSSTRT,7,5,13,0)
006346' 000000  000000
006347' 300000  006355'                     ATABLE  (SSCALL,TS30IN)
006350' 400000  006373'             TS30L:  ATABLE  (SSCHK,TS30NX)
006351' 100000  000000                      ATABLE  (SSSTRT,0,0,0,0)
006352' 000000  000000
006353' 500000  006350'                     ATABLE  (SSJRST,TS30L)
006354' 000000  000000                      ATABLE  (SSLAST)

                                    ; Initialize single step data

006355' 201 00 0 00 000001          TS30IN: MOVEI   1                       ; start with data pattern 1
006356' 202 00 0 00 010022'                 MOVEM   TS30PA#                 ; save it
006357' 476 00 0 00 010023'                 SETOM   TS30SG#                 ; initialize segment number
006360' 201 00 0 00 000016                  MOVEI   16                      ; get initial start address
006361' 202 00 0 00 006424'                 MOVEM   TS30T2                  ;   for 1st subsection
006362' 263 17 0 00 000000                  RTN                             ; return

006363' 402 00 0 00 010005'         TS30A0: SETZM   TDATR                   ; 0 - 0 + cry
006364' 402 00 0 00 010007'                 SETZM   TDATS
006365' 402 00 0 00 010006'                 SETZM   TDATRS
006366' 263 17 0 00 000000                  RTN
```

```

006367' 476 00 0 00 010005'     TS30A1: SETOM   TDATR                   ; -1 - -1 + cry
006370' 476 00 0 00 010007'             SETOM   TDATS
006371' 402 00 0 00 010006'             SETZM   TDATRS
006372' 263 17 0 00 000000              RTN

                                ; Handle single step data

006373' 336 00 0 00 010022'     TS30NX: SKIPN   TS30PA                  ; done with data patterns?
006374' 263 17 0 00 000000              RTN                             ; yes - return +1
006375' 350 01 0 00 010023'             AOS     1,TS30SG                ; point to proper segment
006376' 303 01 0 00 000004              CAILE   1,4                     ; done with this data pattern?
                                        JRST    [MOVE   1,TS30PA        ; yes - make up next one and go
                                                 LSH    1,1             ;   handle that one
                                                 MOVEM  1,TS30PA
                                                 SETOM  TS30SG
                                                 MOVEI  20              ; set up subsequent start address
                                                 MOVEM  TS30T2          ;   for 1st subsection
006377' 254 00 0 00 007664'                      JRST   TS30NX]
006400' 200 02 0 06 000000              MOVE    2,(6)                   ; get table entry
006401' 200 00 0 01 006417'             MOVE    TS30T1(1)               ; get number of single steps
006402' 137 00 0 00 007241'             DPB     [POINT 9,2,11]          ; insert into table entry
006403' 200 00 0 01 006424'             MOVE    TS30T2(1)               ; get starting address
006404' 137 00 0 00 007242'             DPB     [POINT 12,2,23]         ; insert into table entry
006405' 200 00 0 01 006431'             MOVE    TS30T3(1)               ; get ending address
006406' 137 00 0 00 007243'             DPB     [POINT 12,2,35]         ; insert into table entry
006407' 202 02 0 06 000000              MOVEM   2,(6)                   ; save table entry
006410' 402 00 0 00 010005'             SETZM   TDATR                   ; clear initial data
006411' 402 00 0 00 010007'             SETZM   TDATS                   ; clear initial data
006412' 256 00 0 01 006436'             XCT     TS30EB(1)               ; get resulting data pattern
006413' 202 00 0 06 000001              MOVEM   1(6)                    ; save EBUF data
006414' 202 00 0 00 010006'             MOVEM   TDATRS                  ; save result data
006415' 350 00 0 17 000000              AOS     (P)                     ; set up proper return
006416' 263 17 0 00 000000              RTN                             ; return

                                ; Number of single steps

006417' 000000  000004          TS30T1: 4
006420' 000000  000003                  3
006421' 000000  000005                  5
006422' 000000  000003                  3
006423' 000000  000003                  3

                                ; Starting address

006424' 000000  000016          TS30T2: 16
006425' 000000  000024                  24
006426' 000000  000041                  41
006427' 000000  000053                  53
006430' 000000  000056                  56
```

```


                                        ; Ending address

006431' 000000  000023                  TS30T3: 23
006432' 000000  000026                          26
006433' 000000  000045                          45
006434' 000000  000055                          55
006435' 000000  000060                          60

                                        ; Table which calculates the result

                                        TS30EB: GO      [MOVE    1,TS30PA         ; f1 - 0 + cry ==> f1
                                                         MOVEM   1,TDATS
                                                         MOVE    1
006436' 260 17 0 00 007673'                              RTN]
                                                GO      [SETO    0,               ; f0 - 0 + cry ==> 77..7377..77
                                                         MOVE    1,TS30PA
                                                         SETCA   1,
                                                         MOVEM   1,TDATS
                                                         GO      ADDLOG
                                                         MOVEI   1,1
                                                         GO      ADDLOG
006437' 260 17 0 00 007677'                              RTN]
                                                GO      [MOVEI   1                ; f0 - 1 + cry ==> 77..7377..76
                                                         MOVEM   TDATR
                                                         MOVE    1,TS30PA
                                                         SETCA   1,
                                                         MOVEM   1,TDATS
                                                         MOVE    [-2]
                                                         GO      ADDLOG
                                                         MOVEI   1,1
                                                         GO      ADDLOG
006440' 260 17 0 00 007707'                              RTN]
                                                GO      [MOVE    TS30PA           ; f1 - f1 + cry ==> 0
                                                         MOVEM   TDATR
                                                         MOVEM   TDATS
                                                         SETZ
006441' 260 17 0 00 007721'                              RTN]
                                                GO      [MOVE    1,TS30PA         ; f0 - f1 + cry ==> f0 left shifted one
                                                         MOVEM   1,TDATR
                                                         SETCA   1,
                                                         MOVEM   1,TDATS
                                                         MOVE    1
                                                         GO      ADDLOG
                                                         MOVEI   1,1
                                                         GO      ADDLOG
006442' 260 17 0 00 007726'                              RTN]

                                        ; Error messages

006443' 140000  007737'                 MA30:   MSG!TXNOT![ASCIZ /2901 not doing an S-R+CRY function properly/]
006444' 260000  006547'                         CALL!TXALL!RESPNT               ; print S-R data
006445' 000000000000#                           LAST!CALL!TXALL!SSPNT           ; print sequencer data
```

```


                                ;       Do 0 - 0 + cry, verify that this works
006446' 000000  010000          T30M:   MWORD   <ADDR=0,JMAP,J=1,SOA,AND,D=2>                   ; 0
006447' 442000  000040
006450' 000100  000000                  MWORD   <CONT,SOA,SMIN,CRY,D=2,B=17>                    ; 1
006451' 412007  400740
006452' 000200  000000                  MWORD   <CONT,SOA,AND,D=0>                              ; 2
006453' 440000  000340
006454' 000300  000000                  MWORD   <CONT,SOQ,SMIN,CRY,D=2,B=16>                    ; 3
006455' 212007  000740
006456' 000400  042004                  MWORD   <JMAP,J=4,SAB,PLUS,A=16,B=17,D=1,OENA,SELE,MGC=4> ; 4
006457' 101167  405040


                                ;       Do -1 - -1 + cry, verify that this works
006460' 000500  060000                  MWORD   <JMAP,J=6,SOA,AND,D=2>                          ; 5
006461' 442000  000040
006462' 000600  000000                  MWORD   <CONT,SOA,XNOR,B=1,D=2>                         ; 6
006463' 472000  400340
006464' 000700  000000                  MWORD   <CONT,SOA,XNOR,B=17,D=2>                        ; 7
006465' 472007  400340
006466' 001000  000000                  MWORD   <CONT,SOA,XNOR,D=0>                             ; 10
006467' 470000  000340
006470' 001100  000000                  MWORD   <CONT,SAB,SMIN,CRY,D=2,A=1,B=17>                ; 11
006471' 112017  400740
006472' 001200  000000                  MWORD   <CONT,SAQ,SMIN,CRY,D=2,A=1,B=16>                ; 12
006473' 012017  000740
006474' 001300  132004                  MWORD   <JMAP,J=13,SAB,PLUS,D=1,A=16,B=17,OENA,SELE,MGC=4> ; 13
006475' 101167  405040


                                ;       Build a data pattern 0000..001 / Left shift it once
006476' 001600  170000                  MWORD   <ADDR=16,JMAP,J=17,SOA,AND,D=2>                 ; 16 init
006477' 442000  000040
006500' 001700  220000                  MWORD   <JMAP,J=22,SOA,PLUS,D=2,CRY>                    ; 17
006501' 402000  000440
006502' 002000  210000                  MWORD   <JMAP,J=21,SOA,OR,D=1>                          ; 20 left shift
006503' 431000  000040
006504' 002100  000000                  MWORD   <CONT,SOA,OR,D=7>                               ; 21
006505' 437000  000340


                                ;;      Do data pattern (floating 1) - 0 + cry
006506' 002200  000000                  MWORD   <CONT SOA,SMIN,CRY,B=17,D=2>                    ; 22
006507' 412007  400740
006510' 002300  232004                  MWORD   <JMAP,J=23,SOA,OR,A=17,D=1,OENA,SELE,MGC=4>     ; 23
006511' 431170  005040


                                ;;      Do data pattern inverted (floating 0) - 0 + cry
006512' 002400  250000                  MWORD   <JMAP,J=25,SOA,XNOR,B=1,D=2>                    ; 24
006513' 472000  400040
006514' 002500  000000                  MWORD   <CONT,SOA,SMIN,CRY,A=1,B=17,D=2>                ; 25
006515' 412017  400740
```

```
                                        MWORD   <JMAP,J=26,SOA,OR,A=17,D=1,OENA,SELE,MGC=4>     ; 26
```

```
 5380   006516' 002600  262004                  MWORD   <JMAP,J=26,SOA,OR,A=17,D=1,OENA,SELE,MGC=4>     ; 26
 5381   006517' 431170  005040
 5382
 5383                                   ;;      Do data pattern inverted (floating 0) - 1 + cry
 5384
 5385   006520' 004100  420000                  MWORD   <ADDR=41,JMAP,J=42,SOA,XNOR,D=0>                ; 41
 5386   006521' 470000  000040                  MWORD   <CONT,SOB,AND,B=1,D=2>                          ; 42
 5387   006522' 004200  000000                  MWORD   <CONT,SOB,PLUS,B=1,D=2,CRY>                     ; 43
 5388   006523' 342000  400340                  MWORD   <CONT,SAQ,SMIN,CRY,A=1,B=2,D=2>                 ; 44
 5389   006524' 004300  000000                  MWORD   <JMAP,J=45,SOA,OR,A=2,D=1,OENA,SELE,MGC=4>      ; 45
 5390   006525' 302000  400740
 5391   006526' 004400  000000
 5392   006527' 012011  000740
 5393   006530' 004500  452004
 5394   006531' 431020  005040
 5395
 5396                                   ;;      Do f1 - f1 + cry
 5397
 5398   006532' 005300  540000                  MWORD   <ADDR=53,JMAP,J=54,SOA,OR,B=1,D=2>              ; 53
 5399   006533' 432000  400040                  MWORD   <CONT,SAB,SMIN,CRY,B=1,D=2>                     ; 54
 5400   006534' 005400  000000                  MWORD   <JMAP,J=55,SOA,OR,A=1,D=1,OENA,SELE,MGC=4>      ; 55
 5401   006535' 112000  400740
 5402   006536' 005500  552004
 5403   006537' 431010  005040
 5404
 5405                                   ;;      Do f0 - f1 + cry
 5406
 5407   006540' 005600  570000                  MWORD   <JMAP,J=57,SOA,XNOR,D=0>                        ; 56
 5408   006541' 470000  000040                  MWORD   <CONT,SAQ,SMIN,CRY,B=1,D=2>                     ; 57
 5409   006542' 005700  000000                  MWORD   <JMAP,J=60,SOA,OR,A=1,D=1,OENA,SELE,MGC=4>      ; 60
 5410   006543' 012000  400740
 5411   006544' 006000  602004
 5412   006545' 431010  005040
 5413   006546' 777777  777777                  -1
```

```
                                        ;#***********************************************************************
                                        ; Miscellaneous Print Routines
                                        ;#***********************************************************************

006547' 037 00 0 00 007750'     RESPNT: TMSGC   <R:  >
006550' 200 00 0 00 010005'             MOVE    TDATR
006551' 037 13 0 00 000000              PNTHW
006552' 037 00 0 00 007752'             TMSG    <  S:  >
006553' 200 00 0 00 010007'             MOVE    TDATS
006554' 037 13 0 00 000000              PNTHW
006555' 200 01 0 00 000000*             MOVE    1,SCOSW                 ; get switches
006556' 603 01 0 00 000200              TLNE    1,TXTINH                ; full printout?
006557' 037 00 0 00 007754'             TMSGC   <Result (C):  >         ; no
006560' 607 01 0 00 000200              TLNN    1,TXTINH                ; full printout?
006561' 037 00 0 00 007760'             TMSGC   <Result (Correct):  >   ; yes
006562' 200 00 0 00 010006'             MOVE    TDATRS
006563' 037 13 0 00 000000              PNTHW
006564' 603 01 0 00 000200              TLNE    1,TXTINH                ; full printout?
006565' 037 00 0 00 007765'             TMSGC   <       (A):  >         ; no
006566' 607 01 0 00 000200              TLNN    1,TXTINH                ; full printout?
006567' 037 00 0 00 007771'             TMSGC   <       (Actual):  >
006570' 200 00 0 00 002210*             MOVE    AEBUF
006571' 037 13 0 00 000000              PNTHW
006572' 263 17 0 00 000000              RTN
```

```
                                        ;#*******************************************************************
                                        ;* ADDLOG - Routine to logically add 2 numbers
                                        ;
                                        ; This routine gets around the fact that the DADD instruction uses only
                                        ; 35 bits in each word for the magnitude.
                                        ;
                                        ; Arguments:     AC0 - Number A
                                        ;                AC1 - Number B
                                        ;
                                        ; Returns:       AC0 - Result
                                        ;#*******************************************************************

006573' 261 17 0 00 000002              ADDLOG: RPUT    (2,3,4)                 ; save AC's
006576' 200 04 0 00 000000                      MOVE    4,0                     ; free up AC0
006577' 550 02 0 00 000004                      HRRZ    2,4                     ; get right half of A
006600' 550 03 0 00 000001                      HRRZ    3,1                     ; get right half of B
006601' 270 02 0 00 000003                      ADD     2,3                     ; add right halves
006602' 200 00 0 00 000002                      MOVE    0,2                     ; save result
006603' 554 02 0 00 000004                      HLRZ    2,4                     ; get left half of A
006604' 554 03 0 00 000001                      HLRZ    3,1                     ; get left half of B
006605' 270 02 0 00 000003                      ADD     2,3                     ; add right halves
006606' 554 03 0 00 000000                      HLRZ    3,0                     ; get carry from right half
006607' 270 02 0 00 000003                      ADD     2,3                     ; add in carry
006610' 504 00 0 00 000002                      HRL     2                       ; save left half
006611' 262 17 0 00 000004                      RGET    (4,3,2)                 ; restore AC's
006614' 263 17 0 00 000000                      RTN                             ; return
```

```
                                  ;#*********************************************************************
                                  ; End of 2901 Tests (Part 1)
                                  ;#*********************************************************************

                                          XLIST

NO ERRORS DETECTED

PROGRAM BREAK IS 010024
CPU TIME USED 07:43.811

225P CORE USED
```

```
DFPTA4  MACRO %53B(1242) 08:07  1-Jul-85 Page S-1
DFPTA4  MAC       18-Apr-85 10:59          SYMBOL TABLE
```

```
ADDLOG          006573'       SSSTRT   000001  spd  TG12     001367'      TS17     002706'
AEBUF           006570' ext   T10M     001263'      TG13     001451'      TS2      000306'
ALU     020000  000000  spd   T11M     001345'      TG14     002110'      TS20     003163'
BBPNT           000000  ext   T12M     001427'      TG15     002171'      TS21     003454'
BEXEC           006330' ext   T13M     001576'      TG16     002405'      TS22     004017'
CALL    200000  000000  spd   T14M     002141'      TG17     002664'      TS23     004500'
CEBUF           002206' ext   T15M     002267'      TG2      000264'      TS23A0   004522'
E23             000027  spd   T16M     002446'      TG20     003141'      TS23A1   004526'
E24             000030  spd   T17M     002717'      TG21     003432'      TS23EB   004575'
EBUS    400000  000000  spd   T1M      000144'      TG22     003775'      TS23IN   004514'
ERFLG           000015        T20M     003174'      TG23     004456'      TS23L    004507'
GET     262740  000000        T21M     003465'      TG24     004763'      TS23NX   004532'
GO      260740  000000        T22M     004030'      TG25     005272'      TS23PA   010010'
IPACLR          006325' ext   T23M     004605'      TG26     005563'      TS23SG   010011'
LAST    010000  000000  spd   T24M     005114'      TG27     006026'      TS23T1   004556'
MA1             000142'       T25M     005421'      TG3      000647'      TS23T2   004563'
MA10            001260'       T26M     005714'      TG30     006317'      TS23T3   004570'
MA11            001342'       T27M     006155'      TG4      000731'      TS24     005005'
MA12            001424'       T2M      000331'      TG5      001004'      TS24A0   005027'
MA13            001574'       T30M     006446'      TG6      001057'      TS24A1   005034'
MA14            002137'       T3M      000707'      TG7      001141'      TS24EB   005104'
MA15            002264'       T4M      000762'      TL1      000016'      TS24IN   005021'
MA16            002444'       T5M      001035'      TL10     001230'      TS24L    005014'
MA17            002715'       T6M      001117'      TL11     001312'      TS24NX   005041'
MA2             000327'       T7M      001201'      TL12     001374'      TS24PA   010012'
MA20            003172'       TA1      000022'      TL13     001456'      TS24SG   010013'
MA21            003463'       TA10     001234'      TL14     002115'      TS24T1   005065'
MA22            004026'       TA11     001316'      TL15     002176'      TS24T2   005072'
MA23            004602'       TA12     001400'      TL16     002412'      TS24T3   005077'
MA24            005111'       TA13     001462'      TL17     002671'      TS25     005314'
MA25            005416'       TA14     002121'      TL2      000271'      TS25A0   005336'
MA26            005711'       TA15     002202'      TL20     003146'      TS25A1   005342'
MA27            006152'       TA16     002416'      TL21     003437'      TS25EB   005411'
MA3             000704'       TA17     002675'      TL22     004002'      TS25IN   005330'
MA30            006443'       TA2      000275'      TL23     004463'      TS25L    005323'
MA4             000760'       TA20     003152'      TL24     004770'      TS25NX   005346'
MA5             001033'       TA21     003443'      TL25     005277'      TS25PA   010014'
MA6             001114'       TA22     004006'      TL26     005570'      TS25SG   010015'
MA7             001176'       TA23     004467'      TL27     006033'      TS25T1   005372'
MLAST   400000  000000  spd   TA24     004774'      TL3      000654'      TS25T2   005377'
MSG     100000  000000  spd   TA25     005303'      TL30     006324'      TS25T3   005404'
NDMP    000400  000000  spd   TA26     005574'      TL4      000736'      TS26     005605'
P               000017        TA27     006037'      TL5      001011'      TS26A0   005627'
PNTHW   037540  000000        TA3      000660'      TL6      001064'      TS26A1   005634'
PNTMSG  037000  000000        TA30     006330'      TL7      001146'      TS26EB   005704'
PUT     261740  000000        TA4      000742'      TLOAD    006322' ext  TS26IN   005621'
RESPNT          006547'       TA5      001015'      TRACE    006320' ext  TS26L    005614'
RTN     263740  000000        TA6      001070'      TS1      000033'      TS26NX   005641'
SCOPER  027000  000000        TA7      001152'      TS10     001253'      TS26PA   010016'
SCOSW           006555' ext   TDATR    010005'      TS11     001335'      TS26SG   010017'
SSCALL          000003  spd   TDATRS   010006'      TS12     001417'      TS26T1   005665'
SSCHK           000004  spd   TDATS    010007'      TS13     001473'      TS26T2   005672'
SSJRST          000005  spd   TG1      000011'      TS14     002132'      TS26T3   005677'
SSLAST          000000  spd   TG10     001223'      TS15     002221'      TS27     006050'
SSPNT           000000  ext   TG11     001305'      TS16     002427'      TS27A0   006072'
```

```
DFPTA4  MACRO %53B(1242) 08:07  1-Jul-85 Page S-2
DFPTA4  MAC       18-Apr-85 10:59          SYMBOL TABLE
```

```
DFPTA4  MACRO %53B(1242) 08:07  1-Jul-85 Page S-2
DFPTA4  MAC       18-Apr-85 10:59          SYMBOL TABLE

TS27A1          006076'         TSTA5           000773' ent     .LSOA           000000  spd     .RSDO   700000  000000  spd
TS27EB          006145'         TSTA6           001046' ent     .LSOB           000000  spd     .RSELE          005000  spd
TS27IN          006064'         TSTA7           001130' ent     .LSOQ           000000  spd     .RSKCN          240000  spd
TS27L           006057'         TSTSUB          006326' ext     .LSAB           000000  spd     .RSMIN  010000  000000  spd
TS27NX          006102'         TX1             000032'         .LSAQ           000000  spd     .RXNOR  070000  000000  spd
TS27PA          010020'         TX10            001252'         .LSDO           000000  spd     .RXOR   060000  000000  spd
TS27SG          010021'         TX11            001334'         .LSELE          000000  spd
TS27T1          006126'         TX12            001416'         .LSKCN          000000  spd
TS27T2          006133'         TX13            001472'         .LSMIN          000000  spd
TS27T3          006140'         TX14            002131'         .LXNOR          000000  spd
TS2INI          000314'         TX15            002220'         .LXOR           000000  spd
TS2L            000307'         TX16            002426'         .MA             000017  spd
TS2SET          000316'         TX17            002705'         .MAND           000007  spd
TS3             000677'         TX2             000305'         .MB             000017  spd
TS30            006341'         TX20            003162'         .MCONT          000017  spd
TS30A0          006363'         TX21            003453'         .MCRY           000001  spd
TS30A1          006367'         TX22            004016'         .MD             000007  spd
TS30EB          006436'         TX23            004477'         .MJ             007777  spd
TS30IN          006355'         TX24            005004'         .MJMAP          000017  spd
TS30L           006350'         TX25            005313'         .MMGC           001777  spd
TS30NX          006373'         TX26            005604'         .MNAND          000007  spd
TS30PA          010022'         TX27            006047'         .MOENA          000001  spd
TS30SG          010023'         TX3             000676'         .MOR            000007  spd
TS30T1          006417'         TX30            006340'         .MPLUS          000007  spd
TS30T2          006424'         TX4             000752'         .MRMIN          000007  spd
TS30T3          006431'         TX5             001025'         .MSOA           000007  spd
TS4             000753'         TX6             001106'         .MSOB           000007  spd
TS5             001026'         TX7             001170'         .MSOQ           000007  spd
TS6             001107'         TXALL   060000  000000  spd     .MSAB           000007  spd
TS7             001171'         TXNOT   040000  000000  spd     .MSAQ           000007  spd
TSTA1           000000' ent     TXTINH          000200  spd     .MSDO           000007  spd
TSTA10          001212' ent     Z4              000000'         .MSELE          000007  spd
TSTA11          001274' ent     ZALU    000002  000000  spd     .MSKCN          000037  spd
TSTA12          001356' ent     $ARG2           000004          .MSMIN          000007  spd
TSTA13          001440' ent     $B              000043          .MXNOR          000007  spd
TSTA14          002077' ent     $CHR            554743          .MXOR           000007  spd
TSTA15          002160' ent     $GARG           000001          .RA     000010  000000  spd
TSTA16          002374' ent     %ADDR           000061  spd     .RAND   040000  000000  spd
TSTA17          002653' ent     %ML     006000  602004  spd     .RB             400000  spd
TSTA2           000253' ent     %MR     431010  005040  spd     .RCONT          000340  spd
TSTA20          003130' ent     .LA             000000  spd     .RCRY           000400  spd
TSTA21          003421' ent     .LADDR  000100  000000  spd     .RD     001000  000000  spd
TSTA22          003764' ent     .LAND           000000  spd     .RJ             000000  spd
TSTA23          004445' ent     .LB             000000  spd     .RJMAP          000040  spd
TSTA24          004752' ent     .LCONT          000000  spd     .RMGC           000000  spd
TSTA25          005261' ent     .LCRY           000000  spd     .RNAND  050000  000000  spd
TSTA26          005552' ent     .LD             000000  spd     .ROENA          000000  spd
TSTA27          006015' ent     .LJ             010000  spd     .ROR    030000  000000  spd
TSTA3           000636' ent     .LJMAP          000000  spd     .RPLUS          000000  spd
TSTA30          006306' ent     .LMGC           000001  spd     .RRMIN  020000  000000  spd
TSTA31          000000  ext     .LNAND          000000  spd     .RSOA   400000  000000  spd
TSTA32          000000  ext     .LOENA          002000  spd     .RSOB   300000  000000  spd
TSTA33          000000  ext     .LOR            000000  spd     .RSOQ   200000  000000  spd
TSTA34          000000  ext     .LPLUS          000000  spd     .RSAB   100000  000000  spd
TSTA4           000720' ent     .LRMIN          000000  spd     .RSAQ           000000  spd
```

```
ADDLOG    4023   4030   4970   4976   4984   4996   5291   5293   5301   5303   5315   5317   5452#
AEBUF       20#   706    986   1085   1184   1283   1384   1989   5436
                                                                         1352   1460   1855   1957   2180
```

SEQ 1022

SEQ 1035

```
ADDLOG    4023   4030   4970   4976   4984   4996   5291   5293   5301   5303   5315   5317   5452#
AEBUF      20#    706    986   1085   1184   1283   1384   1989   5436
ALU         73    316    674    772    862    954   1053   1152   1251   1352   1460   1855   1957   2180
          2416   2650   2896   3184   3553   3879   4210   4528   4836   5152
BBPNT      20#    192    385    735    825    915   1014   1113   1212   1311   1412   1573   1908   2047
          2241   2471   2705   2951   3239
BEXEC      20#     99    342    700    798    888    980   1079   1178   1277   1378   1486   1881   1983
          2206   2442   2676   2922   3210   3579   3905   4236   4554   4862   5178
CALL       192    385    735    825    915   1014   1113   1212   1311   1412   1573   1908   2047   2241
          2471   2705   2951   3239   3717   3718   4048   4049   4378   4379   4710   4711   5002   5003
          5323   5324
CEBUF      20#    704    984   1083   1182   1281   1382   1987
E23         75    318    676    774    864    956   1055   1154   1253   1354   1462   1857   1959   2182
          2418   2652   2898   3186   3555   3881   4212   4530   4838   5154
E24        676    774    864    956   1055   1154   1253   1354
EBUS        73    316    674    772    862    954   1053   1152   1251   1352   1460   1855   1957   2180
          2416   2650   2896   3184   3553   3879   4210   4528   4836   5152
ERFLG       92    102    109    335    345    352    693    709    716    791    801    808    881    891
           898    973    989    996   1072   1088   1095   1171   1187   1194   1270   1286   1293   1371
          1387   1394   1479   1489   1496   1874   1884   1891   1976   1992   1999   2199   2209   2216
          2435   2445   2452   2669   2679   2686   2915   2925   2932   3203   3213   3220   3572   3582
          3589   3898   3908   3915   4229   4239   4246   4547   4557   4564   4855   4865   4872   5171
          5181   5188
IPACLR     24#     93    336    694    792    882    974   1073   1172   1271   1372   1480   1875   1977
          2200   2436   2670   2916   3204   3573   3899   4230   4548   4856   5172
LAST       192    385    735    825    915   1014   1113   1212   1311   1412   1573   1908   2047   2241
          2471   2705   2951   3239   3718   4049   4379   4711   5003   5324
MA1        106    191#
MA10      1191   1210#
MA11      1290   1309#
MA12      1391   1410#
MA13      1493   1572#
MA14      1888   1907#
MA15      1996   2045#
MA16      2213   2240#
MA17      2449   2470#
MA2        349    384#
MA20      2683   2704#
MA21      2929   2950#
MA22      3217   3238#
MA23      3586   3716#
MA24      3912   4047#
MA25      4243   4377#
MA26      4561   4709#
MA27      4869   5001#
MA3        713    733#
MA30      5185   5322#
MA4        805    824#
MA5        895    914#
MA6        993   1012#
MA7       1092   1111#
MLAST       75    318    676    774    864    956   1055   1154   1253   1354   1462   1857   1959   2182
          2418   2652   2898   3186   3555   3881   4212   4530   4838   5154
```

SEQ 1023

SEQ 1036

```
MSG        191    384    733    734    824    914   1012   1013   1111   1112   1210   1211   1309   1310
          1410   1411   1572   1907   2045   2046   2240   2470   2704   2950   3238   3716   4047   4377
```

SEQ 1023

```
MSG        191   384   733   734   824   914  1012  1013  1111  1112  1210  1211  1309  1310
          1410  1411  1572  1907  2045  2046  2240  2470  2704  2950  3238  3716  4047  4377
          4709  5001  5322
NDMP        73   316   674   772   862   954  1053  1152  1251  1352  1460  1855  1957  2180
          2416  2650  2896  3184  3553  3879  4210  4528  4836  5152
P          374  3655  3983  4312  4632  4938  5254
RESPNT    3717  4048  4378  4710  5002  5323  5419#
SCOSW      20#   28#  5425
SSCALL     360   361  3597  3598  3600  3601  3603  3604  3923  3924  3926  3927  3929  3930
          4254  4255  4257  4258  4260  4261  4572  4573  4575  4576  4578  4579  4880  4881
          4883  4884  4886  4887  5196  5197  5199  5200  5202  5203
SSCHK      361   362  3604  3605  3930  3931  4261  4262  4579  4580  4887  4888  5203  5204
SSJRST     364   365  3607  3608  3933  3934  4264  4265  4582  4583  4890  4891  5206  5207
SSLAST     187   188   365   366   728   729   820   821   910   911  1008  1009  1107  1108
          1206  1207  1305  1306  1406  1407  1568  1569  1903  1904  2041  2042  2236  2237
          2466  2467  2700  2701  2946  2947  3234  3235  3608  3609  3934  3935  4265  4266
          4583  4584  4891  4892  5207  5208
SSPNT      20#  3718  4049  4379  4711  5003  5324
SSSTRT     117   118   119   120   121   122   123   124   125   126   127   128   129   130
           131   132   133   134   135   136   137   138   139   140   141   142   143   144
           145   146   147   148   149   150   151   152   153   154   155   156   157   158
           159   160   161   162   163   164   165   166   167   168   169   170   171   172
           173   174   175   176   177   178   179   180   181   182   183   184   185   186
           187   362   363   364   724   725   726   727   728   816   817   818   819   820
           906   907   908   909   910  1004  1005  1006  1007  1008  1103  1104  1105  1106
          1107  1202  1203  1204  1205  1206  1301  1302  1303  1304  1305  1402  1403  1404
          1405  1406  1504  1505  1506  1507  1508  1509  1510  1511  1512  1513  1514  1515
          1516  1517  1518  1519  1520  1521  1522  1523  1524  1525  1526  1527  1528  1529
          1530  1531  1532  1533  1534  1535  1536  1537  1538  1539  1540  1541  1542  1543
          1544  1545  1546  1547  1548  1549  1550  1551  1552  1553  1554  1555  1556  1557
          1558  1559  1560  1561  1562  1563  1564  1565  1566  1567  1568  1899  1900  1901
          1902  1903  2007  2008  2009  2010  2011  2012  2013  2014  2015  2016  2017  2018
          2019  2020  2021  2022  2023  2024  2025  2026  2027  2028  2029  2030  2031  2032
          2033  2034  2035  2036  2037  2038  2039  2040  2041  2224  2225  2226  2227  2228
          2229  2230  2231  2232  2233  2234  2235  2236  2460  2461  2462  2463  2464  2465
          2466  2694  2695  2696  2697  2698  2699  2700  2940  2941  2942  2943  2944  2945
          2946  3228  3229  3230  3231  3232  3233  3234  3598  3599  3600  3601  3602  3603
          3605  3606  3607  3924  3925  3926  3927  3928  3929  3931  3932  3933  4255  4256
          4257  4258  4259  4260  4262  4263  4264  4573  4574  4575  4576  4577  4578  4580
          4581  4582  4881  4882  4883  4884  4885  4886  4888  4889  4890  5197  5198  5199
          5200  5201  5202  5204  5205  5206
T10M      1153  1165  1216#
T11M      1252  1264  1315#
T12M      1353  1365  1416#
T13M      1461  1473  1577#
T14M      1856  1868  1912#
T15M      1958  1970  2051#
T16M      2181  2193  2245#
T17M      2417  2429  2475#
T1M         74    86   198#
T20M      2651  2663  2709#
T21M      2897  2909  2955#
T22M      3185  3197  3243#
```

SEQ 1036

SEQ 1024

```
T23M      3554  3566  3724#
T24M      3880  3892  4055#
```

SEQ 1037

```
T23M      3554   3566   3724#
T24M      3880   3892   4055#
T25M      4211   4223   4385#
T26M      4529   4541   4717#
T27M      4837   4849   5009#
T2M        317    329    391#
T30M      5153   5165   5328#
T3M        675    687    739#
T4M        773    785    829#
T5M        863    875    919#
T6M        955    967   1018#
T7M       1054   1066   1117#
TA1         99#   101    109
TA10      1178#  1180   1194
TA11      1277#  1279   1293
TA12      1378#  1380   1394
TA13      1486#  1488   1496
TA14      1881#  1883   1891
TA15      1983#  1985   1999
TA16      2206#  2208   2216
TA17      2442#  2444   2452
TA2        342#   344    352
TA20      2676#  2678   2686
TA21      2922#  2924   2932
TA22      3210#  3212   3220
TA23      3579#  3581   3589
TA24      3905#  3907   3915
TA25      4236#  4238   4246
TA26      4554#  4556   4564
TA27      4862#  4864   4872
TA3        700#   702    716
TA30      5178#  5180   5188
TA4        798#   800    808
TA5        888#   890    898
TA6        980#   982    996
TA7       1079#  1081   1095
TDATR     3619   3624   3650#  3650   3693   3703   3708   3945   3951   3978   4027   4033   4039   4276
          4281   4307   4352   4363   4368   4594   4600   4627   4683   4691   4696   4902   4907   4933
          4979   4987   4992   5218   5223   5249   5296   5306   5311   5420
TDATRS    3621   3626   3654#  3654   3947   3948   3953   3954   3982   4278   4283   4311   4596   4597
          4602   4603   4631   4904   4909   4937   5220   5225   5253   5430
TDATS     3620   3625   3651#  3651   3685   3689   3694   3704   3710   3946   3952   3979   4013   4022
          4028   4034   4041   4277   4282   4308   4342   4348   4355   4364   4370   4595   4601   4628
          4662   4673   4682   4692   4698   4903   4908   4934   4968   4975   4982   4988   4994   5219
          5224   5250   5284   5290   5299   5307   5313   5423
TG1         72     84#
TG10      1151   1163#
TG11      1250   1262#
TG12      1351   1363#
TG13      1459   1471#
TG14      1854   1866#
TG15      1956   1968#
TG16      2179   2191#
```

```
TG17      2415   2427#
TG2        315    327#
```

```
TG17        2415    2427#
TG2          315     327#
TG20        2649    2661#
TG21        2895    2907#
TG22        3183    3195#
TG23        3552    3564#
TG24        3878    3890#
TG25        4209    4221#
TG26        4527    4539#
TG27        4835    4847#
TG3          673     685#
TG30        5151    5163#
TG4          771     783#
TG5          861     873#
TG6          953     965#
TG7         1052    1064#
TL1           92#    107
TL10        1171#   1192
TL11        1270#   1291
TL12        1371#   1392
TL13        1479#   1494
TL14        1874#   1889
TL15        1976#   1997
TL16        2199#   2214
TL17        2435#   2450
TL2          335#    350
TL20        2669#   2684
TL21        2915#   2930
TL22        3203#   3218
TL23        3572#   3587
TL24        3898#   3913
TL25        4229#   4244
TL26        4547#   4562
TL27        4855#   4870
TL3          693#    714
TL30        5171#   5186
TL4          791#    806
TL5          881#    896
TL6          973#    994
TL7         1072#   1093
TLOAD         20#     87     330     688     786     876     968    1067    1166    1265    1366    1474    1869    1971
            2194    2430    2664    2910    3198    3567    3893    4224    4542    4850    5166
TRACE         20#     85     328     686     784     874     966    1065    1164    1263    1364    1472    1867    1969
            2192    2428    2662    2908    3196    3565    3891    4222    4540    4848    5164
TS1           95     117#
TS10        1174    1202#
TS11        1273    1301#
TS12        1374    1402#
TS13        1482    1504#
TS14        1877    1899#
TS15        1979    2007#
TS16        2202    2224#
TS17        2438    2460#
```

```
TS2          338     360#
```

```
TS2         338     360#
TS20        2672    2694#
TS21        2918    2940#
TS22        3206    3228#
TS23        3575    3597#
TS23A0      3597    3619#
TS23A1      3600    3624#
TS23EB      3652    3684#
TS23IN      3603    3612#
TS23L       3604#   3607
TS23NX      3604    3631#   3641
TS23PA      3613#   3613    3631    3635    3637    3684    3687    3691    3702    3707
TS23SG      3614#   3614    3633    3638
TS23T1      3643    3660#
TS23T2      3616    3640    3645    3668#
TS23T3      3647    3676#
TS24        3901    3923#
TS24A0      3923    3945#
TS24A1      3926    3951#
TS24EB      3980    4012#
TS24IN      3929    3938#
TS24L       3930#   3933
TS24NX      3930    3959#   3969
TS24PA      3939#   3939    3959    3963    3965    4012    4019    4025    4032    4038
TS24SG      3940#   3940    3961    3966
TS24T1      3971    3988#
TS24T2      3942    3968    3973    3996#
TS24T3      3975    4004#
TS25        4232    4254#
TS25A0      4254    4276#
TS25A1      4257    4281#
TS25EB      4309    4341#
TS25IN      4260    4269#
TS25L       4261#   4264
TS25NX      4261    4288#   4298
TS25PA      4270#   4270    4288    4292    4294    4341    4346    4349    4353    4356    4362    4367    4371
TS25SG      4271#   4271    4290    4295
TS25T1      4300    4317#
TS25T2      4273    4297    4302    4325#
TS25T3      4304    4333#
TS26        4550    4572#
TS26A0      4572    4594#
TS26A1      4575    4600#
TS26EB      4629    4661#
TS26IN      4578    4587#
TS26L       4579#   4582
TS26NX      4579    4608#   4618
TS26PA      4588#   4588    4608    4612    4614    4661    4671    4674    4680    4684    4690    4695    4699
TS26SG      4589#   4589    4610    4615
TS26T1      4620    4637#
TS26T2      4591    4617    4622    4645#
TS26T3      4624    4653#
TS27        4858    4880#
```

```
TS27A0      4880    4902#
```

```
TS27A0     4880    4902#
TS27A1     4883    4907#
TS27EB     4935    4967#
TS27IN     4886    4895#
TS27L      4887#   4890
TS27NX     4887    4914#   4924
TS27PA     4896#   4896    4914    4918    4920    4967    4973    4980    4986    4991
TS27SG     4897#   4897    4916    4921
TS27T1     4926    4943#
TS27T2     4899    4923    4928    4951#
TS27T3     4930    4959#
TS2INI      360     369#
TS2L        361#    364
TS2SET      361     372#
TS3         696     724#
TS30       5174    5196#
TS30A0     5196    5218#
TS30A1     5199    5223#
TS30EB     5251    5283#
TS30IN     5202    5211#
TS30L      5203#   5206
TS30NX     5203    5230#   5240
TS30PA     5212#   5212    5230    5234    5236    5283    5288    5297    5305    5310
TS30SG     5213#   5213    5232    5237
TS30T1     5242    5259#
TS30T2     5215    5239    5244    5267#
TS30T3     5246    5275#
TS4         794     816#
TS5         884     906#
TS6         976    1004#
TS7        1075    1103#
TSTA1        10      72#
TSTA10       10     778     867     958    1056    1151#
TSTA11       11     868     959    1057    1155    1250#
TSTA12       11     960    1058    1156    1254    1351#
TSTA13       11    1059    1157    1255    1355    1459#
TSTA14       11    1158    1256    1356    1463    1854#
TSTA15       11    1257    1357    1464    1858    1956#
TSTA16       11    1358    1465    1859    1960    2179#
TSTA17       11    1466    1860    1961    2183    2415#
TSTA2        10      76     315#
TSTA20       11    1861    1962    2184    2419    2649#
TSTA21       12    1963    2185    2420    2653    2895#
TSTA22       12    2186    2421    2654    2899    3183#
TSTA23       12    2422    2655    2900    3187    3552#
TSTA24       12    2656    2901    3188    3556    3878#
TSTA25       12    2902    3189    3557    3882    4209#
TSTA26       12    3190    3558    3883    4213    4527#
TSTA27       12    3559    3884    4214    4531    4835#
TSTA3        10      77     319     673#
TSTA30       12    3885    4215    4532    4839    5151#
TSTA31       16#   4216    4533    4840    5155
TSTA32       16#   4534    4841    5156
```

ne

```
TSTA33      16#   4842    5157
TSTA34      16#   5158
TSTA4       10      78     320     677     771#
TSTA5       10      79     321     678     775     861#
TSTA6       10     322     679     776     865     953#
TSTA7       10     680     777     866     957    1052#
TSTSUB      20#     94     337     695     793     883     975    1074    1173    1272    1373    1481    1876    1978
          2201    2437    2671    2917    3205    3574    3900    4231    4549    4857    5173
TX1        100     108     113#
TX10      1179    1193    1198#
TX11      1278    1292    1297#
TX12      1379    1393    1398#
TX13      1487    1495    1500#
TX14      1882    1890    1895#
TX15      1984    1998    2003#
TX16      2207    2215    2220#
TX17      2443    2451    2456#
TX2        343     351     356#
TX20      2677    2685    2690#
TX21      2923    2931    2936#
TX22      3211    3219    3224#
TX23      3580    3588    3593#
TX24      3906    3914    3919#
TX25      4237    4245    4250#
TX26      4555    4563    4568#
TX27      4863    4871    4876#
TX3        701     715     720#
TX30      5179    5187    5192#
TX4        799     807     812#
TX5        889     897     902#
TX6        981     995    1000#
TX7       1080    1094    1099#
TXALL      192     385     735     825     915    1014    1113    1212    1311    1412    1573    1908    2047    2241
          2471    2705    2951    3239    3717    3718    4048    4049    4378    4379    4710    4711    5002    5003
          5323    5324
TXNOT      191     384     733     734     824     914    1012    1013    1111    1112    1210    1211    1309    1310
          1410    1411    1572    1907    2045    2046    2240    2470    2704    2950    3238    3716    4047    4377
          4709    5001    5322
TXTINH    5426    5428    5432    5434
Z4          35#     84     327     685     783     873     965    1064    1163    1262    1363    1471    1866    1968
          2191    2427    2661    2907    3195    3564    3890    4221    4539    4847    5163
ZALU        73     316     674     772     862     954    1053    1152    1251    1352    1460    1855    1957    2180
          2416    2650    2896    3184    3553    3879    4210    4528    4836    5152
$ARG2      198#    198     200#    200     202#    202     204#    204     209#    209     211#    211     213#    213
           215#    215     220#    220     222#    222     224#    224     226#    226     231#    231     233#    233
           235#    235     237#    237     242#    242     244#    244     246#    246     248#    248     253#    253
           255#    255     257#    257     259#    259     264#    264     266#    266     268#    268     270#    270
           275#    275     277#    277     279#    279     281#    281     283#    283     285#    285     287#    287
           391#    391     393#    393     395#    395     397#    397     402#    402     404#    404     406#    406
           408#    408     413#    413     415#    415     417#    417     419#    419     424#    424     426#    426
           428#    428     430#    430     435#    435     437#    437     439#    439     441#    441     446#    446
           448#    448     450#    450     452#    452     457#    457     459#    459     461#    461     463#    463
           468#    468     470#    470     472#    472     474#    474     476#    476     478#    478     480#    480
```

```
 482#  482    484#  484    486#  486    488#  488    490#  490    492#  492    494#  494
 499#  499    501#  501    503#  503    505#  505    510#  510    512#  512    514#  514
 516#  516    521#  521    523#  523    525#  525    527#  527    532#  532    534#  534
 536#  536    538#  538    543#  543    545#  545    547#  547    549#  549    554#  554
 556#  556    558#  558    560#  560    565#  565    567#  567    569#  569    571#  571
 576#  576    578#  578    580#  580    582#  582    584#  584    586#  586    588#  588
 590#  590    592#  592    594#  594    596#  596    598#  598    600#  600    602#  602
 607#  607    609#  609    611#  611    613#  613    615#  615    617#  617    619#  619
 621#  621    623#  623    625#  625    627#  627    629#  629    631#  631    633#  633
 739#  739    741#  741    743#  743    745#  745    829#  829    831#  831    833#  833
 835#  835    919#  919    921#  921    923#  923    925#  925   1018# 1018   1020# 1020
1022# 1022   1024# 1024   1117# 1117   1119# 1119   1121# 1121   1123# 1123   1216# 1216
1218# 1218   1220# 1220   1222# 1222   1315# 1315   1317# 1317   1319# 1319   1321# 1321
1416# 1416   1418# 1418   1420# 1420   1422# 1422   1577# 1577   1579# 1579   1581# 1581
1584# 1584   1586# 1586   1588# 1588   1593# 1593   1595# 1595   1597# 1597   1600# 1600
1602# 1602   1604# 1604   1609# 1609   1611# 1611   1613# 1613   1616# 1616   1618# 1618
1620# 1620   1625# 1625   1627# 1627   1629# 1629   1632# 1632   1634# 1634   1636# 1636
1641# 1641   1643# 1643   1645# 1645   1648# 1648   1650# 1650   1652# 1652   1657# 1657
1659# 1659   1661# 1661   1664# 1664   1666# 1666   1668# 1668   1673# 1673   1675# 1675
1677# 1677   1680# 1680   1682# 1682   1684# 1684   1689# 1689   1691# 1691   1693# 1693
1696# 1696   1698# 1698   1700# 1700   1705# 1705   1707# 1707   1709# 1709   1712# 1712
1714# 1714   1716# 1716   1721# 1721   1723# 1723   1725# 1725   1728# 1728   1730# 1730
1732# 1732   1737# 1737   1739# 1739   1741# 1741   1744# 1744   1746# 1746   1748# 1748
1753# 1753   1755# 1755   1757# 1757   1760# 1760   1762# 1762   1764# 1764   1769# 1769
1771# 1771   1773# 1773   1776# 1776   1778# 1778   1780# 1780   1785# 1785   1787# 1787
1789# 1789   1792# 1792   1794# 1794   1796# 1796   1801# 1801   1803# 1803   1805# 1805
1808# 1808   1810# 1810   1812# 1812   1817# 1817   1819# 1819   1821# 1821   1824# 1824
1826# 1826   1828# 1828   1912# 1912   1914# 1914   1916# 1916   1919# 1919   1921# 1921
1923# 1923   1925# 1925   2051# 2051   2053# 2053   2055# 2055   2057# 2057   2059# 2059
2061# 2061   2063# 2063   2065# 2065   2067# 2067   2069# 2069   2071# 2071   2073# 2073
2075# 2075   2077# 2077   2079# 2079   2081# 2081   2083# 2083   2088# 2088   2090# 2090
2092# 2092   2094# 2094   2096# 2096   2098# 2098   2100# 2100   2102# 2102   2104# 2104
2106# 2106   2108# 2108   2110# 2110   2112# 2112   2114# 2114   2116# 2116   2118# 2118
2120# 2120   2245# 2245   2247# 2247   2249# 2249   2251# 2251   2253# 2253   2255# 2255
2257# 2257   2259# 2259   2261# 2261   2263# 2263   2265# 2265   2267# 2267   2269# 2269
2271# 2271   2273# 2273   2275# 2275   2277# 2277   2279# 2279   2281# 2281   2283# 2283
2285# 2285   2287# 2287   2289# 2289   2291# 2291   2293# 2293   2295# 2295   2297# 2297
2299# 2299   2301# 2301   2303# 2303   2305# 2305   2307# 2307   2309# 2309   2311# 2311
2313# 2313   2318# 2318   2320# 2320   2322# 2322   2324# 2324   2326# 2326   2328# 2328
2330# 2330   2332# 2332   2334# 2334   2336# 2336   2338# 2338   2340# 2340   2342# 2342
2344# 2344   2346# 2346   2348# 2348   2350# 2350   2352# 2352   2357# 2357   2359# 2359
2361# 2361   2363# 2363   2365# 2365   2367# 2367   2369# 2369   2371# 2371   2373# 2373
2375# 2375   2377# 2377   2379# 2379   2381# 2381   2475# 2475   2477# 2477   2479# 2479
2481# 2481   2483# 2483   2485# 2485   2487# 2487   2489# 2489   2491# 2491   2493# 2493
2495# 2495   2497# 2497   2499# 2499   2501# 2501   2503# 2503   2505# 2505   2507# 2507
2509# 2509   2511# 2511   2513# 2513   2515# 2515   2517# 2517   2519# 2519   2521# 2521
2523# 2523   2525# 2525   2527# 2527   2529# 2529   2531# 2531   2533# 2533   2535# 2535
2537# 2537   2539# 2539   2541# 2541   2543# 2543   2545# 2545   2550# 2550   2552# 2552
2554# 2554   2556# 2556   2558# 2558   2560# 2560   2562# 2562   2564# 2564   2566# 2566
2568# 2568   2570# 2570   2572# 2572   2574# 2574   2576# 2576   2578# 2578   2580# 2580
2582# 2582   2584# 2584   2586# 2586   2588# 2588   2593# 2593   2595# 2595   2597# 2597
2599# 2599   2601# 2601   2603# 2603   2605# 2605   2607# 2607   2609# 2609   2611# 2611
2613# 2613   2615# 2615   2709# 2709   2711# 2711   2713# 2713   2715# 2715   2717# 2717
```

SEQ 1043

2719#	2719	2721#	2721	2723#	2723	2725#	2725	2727#	2727	2729#	2729	2731#	2731
2733#	2733	2735#	2735	2737#	2737	2739#	2739	2741#	2741	2743#	2743	2745#	2745
2747#	2747	2749#	2749	2751#	2751	2753#	2753	2755#	2755	2757#	2757	2759#	2759
2761#	2761	2763#	2763	2765#	2765	2767#	2767	2769#	2769	2771#	2771	2773#	2773
2775#	2775	2777#	2777	2779#	2779	2784#	2784	2786#	2786	2788#	2788	2790#	2790
2792#	2792	2794#	2794	2796#	2796	2798#	2798	2800#	2800	2802#	2802	2804#	2804
2806#	2806	2808#	2808	2810#	2810	2812#	2812	2814#	2814	2816#	2816	2818#	2818
2820#	2820	2822#	2822	2824#	2824	2826#	2826	2828#	2828	2830#	2830	2835#	2835
2837#	2837	2839#	2839	2841#	2841	2843#	2843	2845#	2845	2847#	2847	2849#	2849
2851#	2851	2853#	2853	2855#	2855	2857#	2857	2859#	2859	2861#	2861	2955#	2955
2957#	2957	2959#	2959	2961#	2961	2963#	2963	2965#	2965	2967#	2967	2969#	2969
2971#	2971	2973#	2973	2975#	2975	2977#	2977	2979#	2979	2981#	2981	2983#	2983
2985#	2985	2987#	2987	2989#	2989	2991#	2991	2993#	2993	2995#	2995	2997#	2997
2999#	2999	3001#	3001	3003#	3003	3008#	3008	3010#	3010	3012#	3012	3014#	3014
3016#	3016	3018#	3018	3020#	3020	3022#	3022	3024#	3024	3026#	3026	3028#	3028
3030#	3030	3032#	3032	3034#	3034	3036#	3036	3038#	3038	3040#	3040	3042#	3042
3044#	3044	3046#	3046	3048#	3048	3050#	3050	3052#	3052	3054#	3054	3056#	3056
3058#	3058	3060#	3060	3062#	3062	3064#	3064	3066#	3066	3071#	3071	3073#	3073
3075#	3075	3077#	3077	3079#	3079	3081#	3081	3083#	3083	3085#	3085	3087#	3087
3089#	3089	3091#	3091	3093#	3093	3095#	3095	3097#	3097	3099#	3099	3101#	3101
3103#	3103	3105#	3105	3107#	3107	3109#	3109	3111#	3111	3113#	3113	3115#	3115
3117#	3117	3119#	3119	3121#	3121	3123#	3123	3125#	3125	3127#	3127	3129#	3129
3131#	3131	3133#	3133	3135#	3135	3137#	3137	3139#	3139	3141#	3141	3143#	3143
3145#	3145	3147#	3147	3149#	3149	3243#	3243	3245#	3245	3247#	3247	3250#	3250
3252#	3252	3254#	3254	3256#	3256	3258#	3258	3260#	3260	3262#	3262	3264#	3264
3266#	3266	3268#	3268	3270#	3270	3272#	3272	3274#	3274	3276#	3276	3278#	3278
3280#	3280	3282#	3282	3284#	3284	3286#	3286	3288#	3288	3290#	3290	3292#	3292
3294#	3294	3296#	3296	3298#	3298	3300#	3300	3302#	3302	3304#	3304	3306#	3306
3308#	3308	3310#	3310	3312#	3312	3314#	3314	3316#	3316	3318#	3318	3320#	3320
3325#	3325	3327#	3327	3329#	3329	3331#	3331	3333#	3333	3335#	3335	3337#	3337
3339#	3339	3341#	3341	3343#	3343	3345#	3345	3347#	3347	3349#	3349	3351#	3351
3353#	3353	3355#	3355	3357#	3357	3359#	3359	3361#	3361	3363#	3363	3365#	3365
3367#	3367	3369#	3369	3371#	3371	3373#	3373	3375#	3375	3377#	3377	3379#	3379
3384#	3384	3386#	3386	3388#	3388	3390#	3390	3392#	3392	3394#	3394	3396#	3396
3398#	3398	3400#	3400	3402#	3402	3404#	3404	3406#	3406	3408#	3408	3410#	3410
3412#	3412	3414#	3414	3416#	3416	3418#	3418	3420#	3420	3422#	3422	3424#	3424
3426#	3426	3428#	3428	3430#	3430	3432#	3432	3434#	3434	3436#	3436	3438#	3438
3440#	3440	3442#	3442	3444#	3444	3446#	3446	3448#	3448	3450#	3450	3452#	3452
3454#	3454	3456#	3456	3458#	3458	3460#	3460	3462#	3462	3464#	3464	3466#	3466
3468#	3468	3470#	3470	3472#	3472	3474#	3474	3476#	3476	3478#	3478	3480#	3480
3482#	3482	3484#	3484	3486#	3486	3488#	3488	3490#	3490	3492#	3492	3494#	3494
3496#	3496	3498#	3498	3500#	3500	3502#	3502	3504#	3504	3506#	3506	3508#	3508
3510#	3510	3512#	3512	3514#	3514	3516#	3516	3724#	3724	3726#	3726	3728#	3728
3730#	3730	3732#	3732	3734#	3734	3736#	3736	3738#	3738	3740#	3740	3742#	3742
3747#	3747	3749#	3749	3751#	3751	3753#	3753	3755#	3755	3757#	3757	3759#	3759
3761#	3761	3766#	3766	3768#	3768	3770#	3770	3772#	3772	3777#	3777	3779#	3779
3781#	3781	3783#	3783	3785#	3785	3787#	3787	3789#	3789	3791#	3791	3796#	3796
3798#	3798	3800#	3800	3802#	3802	3804#	3804	3806#	3806	3808#	3808	3810#	3810
3812#	3812	3817#	3817	3819#	3819	3821#	3821	3823#	3823	3825#	3825	3830#	3830
3832#	3832	3834#	3834	3839#	3839	3841#	3841	3843#	3843	4055#	4055	4057#	4057
4059#	4059	4061#	4061	4063#	4063	4065#	4065	4067#	4067	4069#	4069	4071#	4071
4073#	4073	4078#	4078	4080#	4080	4082#	4082	4084#	4084	4086#	4086	4088#	4088
4090#	4090	4092#	4092	4097#	4097	4099#	4099	4101#	4101	4103#	4103	4108#	4108

```
    4110#  4110   4112#  4112   4114#  4114   4116#  4116   4118#  4118   4120#  4120   4122#  4122
    4127#  4127   4129#  4129   4131#  4131   4133#  4133   4135#  4135   4137#  4137   4139#  4139      SEQ 1044
    4141#  4141   4143#  4143   4148#  4148   4150#  4150   4152#  4152   4154#  4154   4156#  4156
    4161#  4161   4163#  4163   4165#  4165   4170#  4170   4172#  4172   4174#  4174   4385#  4385
    4387#  4387   4389#  4389   4391#  4391   4393#  4393   4395#  4395   4397#  4397   4402#  4402
    4404#  4404   4406#  4406   4408#  4408   4410#  4410   4412#  4412   4414#  4414   4419#  4419
    4421#  4421   4423#  4423   4425#  4425   4430#  4430   4432#  4432   4434#  4434   4436#  4436
    4438#  4438   4440#  4440   4442#  4442   4447#  4447   4449#  4449   4451#  4451   4453#  4453
    4455#  4455   4457#  4457   4459#  4459   4461#  4461   4466#  4466   4468#  4468   4470#  4470
    4472#  4472   4474#  4474   4479#  4479   4481#  4481   4483#  4483   4488#  4488   4490#  4490
    4492#  4492   4717#  4717   4719#  4719   4721#  4721   4723#  4723   4725#  4725   4730#  4730
    4732#  4732   4734#  4734   4736#  4736   4738#  4738   4740#  4740   4742#  4742   4747#  4747
    4749#  4749   4751#  4751   4753#  4753   4758#  4758   4760#  4760   4765#  4765   4767#  4767
    4769#  4769   4774#  4774   4776#  4776   4778#  4778   4780#  4780   4782#  4782   4787#  4787
    4789#  4789   4791#  4791   4796#  4796   4798#  4798   4800#  4800   5009#  5009   5011#  5011
    5013#  5013   5015#  5015   5017#  5017   5019#  5019   5021#  5021   5026#  5026   5028#  5028
    5030#  5030   5032#  5032   5034#  5034   5036#  5036   5038#  5038   5043#  5043   5045#  5045
    5047#  5047   5049#  5049   5054#  5054   5056#  5056   5058#  5058   5060#  5060   5062#  5062
    5064#  5064   5066#  5066   5071#  5071   5073#  5073   5075#  5075   5077#  5077   5079#  5079
    5081#  5081   5083#  5083   5085#  5085   5090#  5090   5092#  5092   5094#  5094   5096#  5096
    5098#  5098   5103#  5103   5105#  5105   5107#  5107   5112#  5112   5114#  5114   5116#  5116
    5328#  5328   5330#  5330   5332#  5332   5334#  5334   5336#  5336   5341#  5341   5343#  5343
    5345#  5345   5347#  5347   5349#  5349   5351#  5351   5353#  5353   5358#  5358   5360#  5360
    5362#  5362   5364#  5364   5369#  5369   5371#  5371   5376#  5376   5378#  5378   5380#  5380
    5385#  5385   5387#  5387   5389#  5389   5391#  5391   5393#  5393   5398#  5398   5400#  5400
    5402#  5402   5407#  5407   5409#  5409   5411#  5411
$B   198#   198    200#   200    202#   202    204#   204    209#   209    211#   211    213#   213
     215#   215    220#   220    222#   222    224#   224    226#   226    231#   231    233#   233
     235#   235    237#   237    242#   242    244#   244    246#   246    248#   248    253#   253
     255#   255    257#   257    259#   259    264#   264    266#   266    268#   268    270#   270
     275#   275    277#   277    279#   279    281#   281    283#   283    285#   285    287#   287
     391#   391    393#   393    395#   395    397#   397    402#   402    404#   404    406#   406
     408#   408    413#   413    415#   415    417#   417    419#   419    424#   424    426#   426
     428#   428    430#   430    435#   435    437#   437    439#   439    441#   441    446#   446
     448#   448    450#   450    452#   452    457#   457    459#   459    461#   461    463#   463
     468#   468    470#   470    472#   472    474#   474    476#   476    478#   478    480#   480
     482#   482    484#   484    486#   486    488#   488    490#   490    492#   492    494#   494
     499#   499    501#   501    503#   503    505#   505    510#   510    512#   512    514#   514
     516#   516    521#   521    523#   523    525#   525    527#   527    532#   532    534#   534
     536#   536    538#   538    543#   543    545#   545    547#   547    549#   549    554#   554
     556#   556    558#   558    560#   560    565#   565    567#   567    569#   569    571#   571
     576#   576    578#   578    580#   580    582#   582    584#   584    586#   586    588#   588
     590#   590    592#   592    594#   594    596#   596    598#   598    600#   600    602#   602
     607#   607    609#   609    611#   611    613#   613    615#   615    617#   617    619#   619
     621#   621    623#   623    625#   625    627#   627    629#   629    631#   631    633#   633
     739#   739    741#   741    743#   743    745#   745    829#   829    831#   831    833#   833
     835#   835    919#   919    921#   921    923#   923    925#   925   1018#  1018   1020#  1020
    1022#  1022   1024#  1024   1117#  1117   1119#  1119   1121#  1121   1123#  1123   1216#  1216
    1218#  1218   1220#  1220   1222#  1222   1315#  1315   1317#  1317   1319#  1319   1321#  1321
    1416#  1416   1418#  1418   1420#  1420   1422#  1422   1577#  1577   1579#  1579   1581#  1581
    1584#  1584   1586#  1586   1588#  1588   1593#  1593   1595#  1595   1597#  1597   1600#  1600
    1602#  1602   1604#  1604   1609#  1609   1611#  1611   1613#  1613   1616#  1616   1618#  1618
    1620#  1620   1625#  1625   1627#  1627   1629#  1629   1632#  1632   1634#  1634   1636#  1636
```

```
1641#  1641   1643#  1643   1645#  1645   1648#  1648   1650#  1650   1652#  1652   1657#  1657     SEQ 1045
1659#  1659   1661#  1661   1664#  1664   1666#  1666   1668#  1668   1673#  1673   1675#  1675
1677#  1677   1680#  1680   1682#  1682   1684#  1684   1689#  1689   1691#  1691   1693#  1693
1696#  1696   1698#  1698   1700#  1700   1705#  1705   1707#  1707   1709#  1709   1712#  1712
1714#  1714   1716#  1716   1721#  1721   1723#  1723   1725#  1725   1728#  1728   1730#  1730
1732#  1732   1737#  1737   1739#  1739   1741#  1741   1744#  1744   1746#  1746   1748#  1748
1753#  1753   1755#  1755   1757#  1757   1760#  1760   1762#  1762   1764#  1764   1769#  1769
1771#  1771   1773#  1773   1776#  1776   1778#  1778   1780#  1780   1785#  1785   1787#  1787
1789#  1789   1792#  1792   1794#  1794   1796#  1796   1801#  1801   1803#  1803   1805#  1805
1808#  1808   1810#  1810   1812#  1812   1817#  1817   1819#  1819   1821#  1821   1824#  1824
1826#  1826   1828#  1828   1912#  1912   1914#  1914   1916#  1916   1919#  1919   1921#  1921
1923#  1923   1925#  1925   2051#  2051   2053#  2053   2055#  2055   2057#  2057   2059#  2059
2061#  2061   2063#  2063   2065#  2065   2067#  2067   2069#  2069   2071#  2071   2073#  2073
2075#  2075   2077#  2077   2079#  2079   2081#  2081   2083#  2083   2088#  2088   2090#  2090
2092#  2092   2094#  2094   2096#  2096   2098#  2098   2100#  2100   2102#  2102   2104#  2104
2106#  2106   2108#  2108   2110#  2110   2112#  2112   2114#  2114   2116#  2116   2118#  2118
2120#  2120   2245#  2245   2247#  2247   2249#  2249   2251#  2251   2253#  2253   2255#  2255
2257#  2257   2259#  2259   2261#  2261   2263#  2263   2265#  2265   2267#  2267   2269#  2269
2271#  2271   2273#  2273   2275#  2275   2277#  2277   2279#  2279   2281#  2281   2283#  2283
2285#  2285   2287#  2287   2289#  2289   2291#  2291   2293#  2293   2295#  2295   2297#  2297
2299#  2299   2301#  2301   2303#  2303   2305#  2305   2307#  2307   2309#  2309   2311#  2311
2313#  2313   2318#  2318   2320#  2320   2322#  2322   2324#  2324   2326#  2326   2328#  2328
2330#  2330   2332#  2332   2334#  2334   2336#  2336   2338#  2338   2340#  2340   2342#  2342
2344#  2344   2346#  2346   2348#  2348   2350#  2350   2352#  2352   2357#  2357   2359#  2359
2361#  2361   2363#  2363   2365#  2365   2367#  2367   2369#  2369   2371#  2371   2373#  2373
2375#  2375   2377#  2377   2379#  2379   2381#  2381   2475#  2475   2477#  2477   2479#  2479
2481#  2481   2483#  2483   2485#  2485   2487#  2487   2489#  2489   2491#  2491   2493#  2493
2495#  2495   2497#  2497   2499#  2499   2501#  2501   2503#  2503   2505#  2505   2507#  2507
2509#  2509   2511#  2511   2513#  2513   2515#  2515   2517#  2517   2519#  2519   2521#  2521
2523#  2523   2525#  2525   2527#  2527   2529#  2529   2531#  2531   2533#  2533   2535#  2535
2537#  2537   2539#  2539   2541#  2541   2543#  2543   2545#  2545   2550#  2550   2552#  2552
2554#  2554   2556#  2556   2558#  2558   2560#  2560   2562#  2562   2564#  2564   2566#  2566
2568#  2568   2570#  2570   2572#  2572   2574#  2574   2576#  2576   2578#  2578   2580#  2580
2582#  2582   2584#  2584   2586#  2586   2588#  2588   2593#  2593   2595#  2595   2597#  2597
2599#  2599   2601#  2601   2603#  2603   2605#  2605   2607#  2607   2609#  2609   2611#  2611
2613#  2613   2615#  2615   2709#  2709   2711#  2711   2713#  2713   2715#  2715   2717#  2717
2719#  2719   2721#  2721   2723#  2723   2725#  2725   2727#  2727   2729#  2729   2731#  2731
2733#  2733   2735#  2735   2737#  2737   2739#  2739   2741#  2741   2743#  2743   2745#  2745
2747#  2747   2749#  2749   2751#  2751   2753#  2753   2755#  2755   2757#  2757   2759#  2759
2761#  2761   2763#  2763   2765#  2765   2767#  2767   2769#  2769   2771#  2771   2773#  2773
2775#  2775   2777#  2777   2779#  2779   2784#  2784   2786#  2786   2788#  2788   2790#  2790
2792#  2792   2794#  2794   2796#  2796   2798#  2798   2800#  2800   2802#  2802   2804#  2804
2806#  2806   2808#  2808   2810#  2810   2812#  2812   2814#  2814   2816#  2816   2818#  2818
2820#  2820   2822#  2822   2824#  2824   2826#  2826   2828#  2828   2830#  2830   2835#  2835
2837#  2837   2839#  2839   2841#  2841   2843#  2843   2845#  2845   2847#  2847   2849#  2849
2851#  2851   2853#  2853   2855#  2855   2857#  2857   2859#  2859   2861#  2861   2955#  2955
2957#  2957   2959#  2959   2961#  2961   2963#  2963   2965#  2965   2967#  2967   2969#  2969
2971#  2971   2973#  2973   2975#  2975   2977#  2977   2979#  2979   2981#  2981   2983#  2983
2985#  2985   2987#  2987   2989#  2989   2991#  2991   2993#  2993   2995#  2995   2997#  2997
2999#  2999   3001#  3001   3003#  3003   3008#  3008   3010#  3010   3012#  3012   3014#  3014
3016#  3016   3018#  3018   3020#  3020   3022#  3022   3024#  3024   3026#  3026   3028#  3028
3030#  3030   3032#  3032   3034#  3034   3036#  3036   3038#  3038   3040#  3040   3042#  3042
3044#  3044   3046#  3046   3048#  3048   3050#  3050   3052#  3052   3054#  3054   3056#  3056
```

```
3058#  3058   3060#  3060   3062#  3062   3064#  3064   3066#  3066   3071#  3071   3073#  3073
3075#  3075   3077#  3077   3079#  3079   3081#  3081   3083#  3083   3085#  3085   3087#  3087
3089#  3089   3091#  3091   3093#  3093   3095#  3095   3097#  3097   3099#  3099   3101#  3101
3103#  3103   3105#  3105   3107#  3107   3109#  3109   3111#  3111   3113#  3113   3115#  3115
3117#  3117   3119#  3119   3121#  3121   3123#  3123   3125#  3125   3127#  3127   3129#  3129
3131#  3131   3133#  3133   3135#  3135   3137#  3137   3139#  3139   3141#  3141   3143#  3143
3145#  3145   3147#  3147   3149#  3149   3243#  3243   3245#  3245   3247#  3247   3250#  3250
3252#  3252   3254#  3254   3256#  3256   3258#  3258   3260#  3260   3262#  3262   3264#  3264
3266#  3266   3268#  3268   3270#  3270   3272#  3272   3274#  3274   3276#  3276   3278#  3278
3280#  3280   3282#  3282   3284#  3284   3286#  3286   3288#  3288   3290#  3290   3292#  3292
3294#  3294   3296#  3296   3298#  3298   3300#  3300   3302#  3302   3304#  3304   3306#  3306
3308#  3308   3310#  3310   3312#  3312   3314#  3314   3316#  3316   3318#  3318   3320#  3320
3325#  3325   3327#  3327   3329#  3329   3331#  3331   3333#  3333   3335#  3335   3337#  3337
3339#  3339   3341#  3341   3343#  3343   3345#  3345   3347#  3347   3349#  3349   3351#  3351
3353#  3353   3355#  3355   3357#  3357   3359#  3359   3361#  3361   3363#  3363   3365#  3365
3367#  3367   3369#  3369   3371#  3371   3373#  3373   3375#  3375   3377#  3377   3379#  3379
3384#  3384   3386#  3386   3388#  3388   3390#  3390   3392#  3392   3394#  3394   3396#  3396
3398#  3398   3400#  3400   3402#  3402   3404#  3404   3406#  3406   3408#  3408   3410#  3410
3412#  3412   3414#  3414   3416#  3416   3418#  3418   3420#  3420   3422#  3422   3424#  3424
3426#  3426   3428#  3428   3430#  3430   3432#  3432   3434#  3434   3436#  3436   3438#  3438
3440#  3440   3442#  3442   3444#  3444   3446#  3446   3448#  3448   3450#  3450   3452#  3452
3454#  3454   3456#  3456   3458#  3458   3460#  3460   3462#  3462   3464#  3464   3466#  3466
3468#  3468   3470#  3470   3472#  3472   3474#  3474   3476#  3476   3478#  3478   3480#  3480
3482#  3482   3484#  3484   3486#  3486   3488#  3488   3490#  3490   3492#  3492   3494#  3494
3496#  3496   3498#  3498   3500#  3500   3502#  3502   3504#  3504   3506#  3506   3508#  3508
3510#  3510   3512#  3512   3514#  3514   3516#  3516   3724#  3724   3726#  3726   3728#  3728
3730#  3730   3732#  3732   3734#  3734   3736#  3736   3738#  3738   3740#  3740   3742#  3742
3747#  3747   3749#  3749   3751#  3751   3753#  3753   3755#  3755   3757#  3757   3759#  3759
3761#  3761   3766#  3766   3768#  3768   3770#  3770   3772#  3772   3777#  3777   3779#  3779
3781#  3781   3783#  3783   3785#  3785   3787#  3787   3789#  3789   3791#  3791   3796#  3796
3798#  3798   3800#  3800   3802#  3802   3804#  3804   3806#  3806   3808#  3808   3810#  3810
3812#  3812   3817#  3817   3819#  3819   3821#  3821   3823#  3823   3825#  3825   3830#  3830
3832#  3832   3834#  3834   3839#  3839   3841#  3841   3843#  3843   4055#  4055   4057#  4057
4059#  4059   4061#  4061   4063#  4063   4065#  4065   4067#  4067   4069#  4069   4071#  4071
4073#  4073   4078#  4078   4080#  4080   4082#  4082   4084#  4084   4086#  4086   4088#  4088
4090#  4090   4092#  4092   4097#  4097   4099#  4099   4101#  4101   4103#  4103   4108#  4108
4110#  4110   4112#  4112   4114#  4114   4116#  4116   4118#  4118   4120#  4120   4122#  4122
4127#  4127   4129#  4129   4131#  4131   4133#  4133   4135#  4135   4137#  4137   4139#  4139
4141#  4141   4143#  4143   4148#  4148   4150#  4150   4152#  4152   4154#  4154   4156#  4156
4161#  4161   4163#  4163   4165#  4165   4170#  4170   4172#  4172   4174#  4174   4385#  4385
4387#  4387   4389#  4389   4391#  4391   4393#  4393   4395#  4395   4397#  4397   4402#  4402
4404#  4404   4406#  4406   4408#  4408   4410#  4410   4412#  4412   4414#  4414   4419#  4419
4421#  4421   4423#  4423   4425#  4425   4430#  4430   4432#  4432   4434#  5434   4436#  4436
4438#  4438   4440#  4440   4442#  4442   4447#  4447   4449#  4449   4451#  4451   4453#  4453
4455#  4455   4457#  4457   4459#  4459   4461#  4461   4466#  4466   4468#  4468   4470#  4470
4472#  4472   4474#  4474   4479#  4479   4481#  4481   4483#  4483   4488#  4488   4490#  4490
4492#  4492   4717#  4717   4719#  4719   4721#  4721   4723#  4723   4725#  4725   4730#  4730
4732#  4732   4734#  4734   4736#  4736   4738#  4738   4740#  4740   4742#  4742   4747#  4747
4749#  4749   4751#  4751   4753#  4753   4758#  4758   4760#  4760   4765#  4765   4767#  4767
4769#  4769   4774#  4774   4776#  4776   4778#  4778   4780#  4780   4782#  4782   4787#  4787
4789#  4789   4791#  4791   4796#  4796   4798#  4798   4800#  4800   5009#  5009   5011#  5011
5013#  5013   5015#  5015   5017#  5017   5019#  5019   5021#  5021   5026#  5026   5028#  5028
5030#  5030   5032#  5032   5034#  5034   5036#  5036   5038#  5038   5043#  5043   5045#  5045
```

```
      5047# 5047   5049# 5049   5054# 5054   5056# 5056   5058# 5058   5060# 5060   5062# 5062   SEQ 1047
      5064# 5064   5066# 5066   5071# 5071   5073# 5073   5075# 5075   5077# 5077   5079# 5079
      5081# 5081   5083# 5083   5085# 5085   5090# 5090   5092# 5092   5094# 5094   5096# 5096
      5098# 5098   5103# 5103   5105# 5105   5107# 5107   5112# 5112   5114# 5114   5116# 5116
      5328# 5328   5330# 5330   5332# 5332   5334# 5334   5336# 5336   5341# 5341   5343# 5343
      5345# 5345   5347# 5347   5349# 5349   5351# 5351   5353# 5353   5358# 5358   5360# 5360
      5362# 5362   5364# 5364   5369# 5369   5371# 5371   5376# 5376   5378# 5378   5380# 5380
      5385# 5385   5387# 5387   5389# 5389   5391# 5391   5393# 5393   5398# 5398   5400# 5400
      5402# 5402   5407# 5407   5409# 5409   5411# 5411
$CHR   198#  198    200#  200    202#  202    204#  204    209#  209    211#  211    213#  213
       215#  215    220#  220    222#  222    224#  224    226#  226    231#  231    233#  233
       235#  235    237#  237    242#  242    244#  244    246#  246    248#  248    253#  253
       255#  255    257#  257    259#  259    264#  264    266#  266    268#  268    270#  270
       275#  275    277#  277    279#  279    281#  281    283#  283    285#  285    287#  287
       391#  391    393#  393    395#  395    397#  397    402#  402    404#  404    406#  406
       408#  408    413#  413    415#  415    417#  417    419#  419    424#  424    426#  426
       428#  428    430#  430    435#  435    437#  437    439#  439    441#  441    446#  446
       448#  448    450#  450    452#  452    457#  457    459#  459    461#  461    463#  463
       468#  468    470#  470    472#  472    474#  474    476#  476    478#  478    480#  480
       482#  482    484#  484    486#  486    488#  488    490#  490    492#  492    494#  494
       499#  499    501#  501    503#  503    505#  505    510#  510    512#  512    514#  514
       516#  516    521#  521    523#  523    525#  525    527#  527    532#  532    534#  534
       536#  536    538#  538    543#  543    545#  545    547#  547    549#  549    554#  554
       556#  556    558#  558    560#  560    565#  565    567#  567    569#  569    571#  571
       576#  576    578#  578    580#  580    582#  582    584#  584    586#  586    588#  588
       590#  590    592#  592    594#  594    596#  596    598#  598    600#  600    602#  602
       607#  607    609#  609    611#  611    613#  613    615#  615    617#  617    619#  619
       621#  621    623#  623    625#  625    627#  627    629#  629    631#  631    633#  633
       739#  739    741#  741    743#  743    745#  745    829#  829    831#  831    833#  833
       835#  835    919#  919    921#  921    923#  923    925#  925   1018# 1018   1020# 1020
      1022# 1022   1024# 1024   1117# 1117   1119# 1119   1121# 1121   1123# 1123   1216# 1216   .LA
      1218# 1218   1220# 1220   1222# 1222   1315# 1315   1317# 1317   1319# 1319   1321# 1321
      1416# 1416   1418# 1418   1420# 1420   1422# 1422   1577# 1577   1579# 1579   1581# 1581
      1584# 1584   1586# 1586   1588# 1588   1593# 1593   1595# 1595   1597# 1597   1600# 1600
      1602# 1602   1604# 1604   1609# 1609   1611# 1611   1613# 1613   1616# 1616   1618# 1618
      1620# 1620   1625# 1625   1627# 1627   1629# 1629   1632# 1632   1634# 1634   1636# 1636
      1641# 1641   1643# 1643   1645# 1645   1648# 1648   1650# 1650   1652# 1652   1657# 1657
      1659# 1659   1661# 1661   1664# 1664   1666# 1666   1668# 1668   1673# 1673   1675# 1675
      1677# 1677   1680# 1680   1682# 1682   1684# 1684   1689# 1689   1691# 1691   1693# 1693
      1696# 1696   1698# 1698   1700# 1700   1705# 1705   1707# 1707   1709# 1709   1712# 1712
      1714# 1714   1716# 1716   1721# 1721   1723# 1723   1725# 1725   1728# 1728   1730# 1730
      1732# 1732   1737# 1737   1739# 1739   1741# 1741   1744# 1744   1746# 1746   1748# 1748
      1753# 1753   1755# 1755   1757# 1757   1760# 1760   1762# 1762   1764# 1764   1769# 1769
      1771# 1771   1773# 1773   1776# 1776   1778# 1778   1780# 1780   1785# 1785   1787# 1787
      1789# 1789   1792# 1792   1794# 1794   1796# 1796   1801# 1801   1803# 1803   1805# 1805
      1808# 1808   1810# 1810   1812# 1812   1817# 1817   1819# 1819   1821# 1821   1824# 1824
      1826# 1826   1828# 1828   1912# 1912   1914# 1914   1916# 1916   1919# 1919   1921# 1921
      1923# 1923   1925# 1925   2051# 2051   2053# 2053   2055# 2055   2057# 2057   2059# 2059
      2061# 2061   2063# 2063   2065# 2065   2067# 2067   2069# 2069   2071# 2071   2073# 2073
      2075# 2075   2077# 2077   2079# 2079   2081# 2081   2083# 2083   2088# 2088   2090# 2090
      2092# 2092   2094# 2094   2096# 2096   2098# 2098   2100# 2100   2102# 2102   2104# 2104
      2106# 2106   2108# 2108   2110# 2110   2112# 2112   2114# 2114   2116# 2116   2118# 2118
      2120# 2120   2245# 2245   2247# 2247   2249# 2249   2251# 2251   2253# 2253   2255# 2255
```

```
2257#  2257    2259#  2259    2261#  2261    2263#  2263    2265#  2265    2267#  2267    2269#  2269        SEQ 1048
2271#  2271    2273#  2273    2275#  2275    2277#  2277    2279#  2279    2281#  2281    2283#  2283
2285#  2285    2287#  2287    2289#  2289    2291#  2291    2293#  2293    2295#  2295    2297#  2297
2299#  2299    2301#  2301    2303#  2303    2305#  2305    2307#  2307    2309#  2309    2311#  2311
2313#  2313    2318#  2318    2320#  2320    2322#  2322    2324#  2324    2326#  2326    2328#  2328
2330#  2330    2332#  2332    2334#  2334    2336#  2336    2338#  2338    2340#  2340    2342#  2342
2344#  2344    2346#  2346    2348#  2348    2350#  2350    2352#  2352    2357#  2357    2359#  2359
2361#  2361    2363#  2363    2365#  2365    2367#  2367    2369#  2369    2371#  2371    2373#  2373
2375#  2375    2377#  2377    2379#  2379    2381#  2381    2475#  2475    2477#  2477    2479#  2479
2481#  2481    2483#  2483    2485#  2485    2487#  2487    2489#  2489    2491#  2491    2493#  2493
2495#  2495    2497#  2497    2499#  2499    2501#  2501    2503#  2503    2505#  2505    2507#  2507
2509#  2509    2511#  2511    2513#  2513    2515#  2515    2517#  2517    2519#  2519    2521#  2521
2523#  2523    2525#  2525    2527#  2527    2529#  2529    2531#  2531    2533#  2533    2535#  2535
2537#  2537    2539#  2539    2541#  2541    2543#  2543    2545#  2545    2550#  2550    2552#  2552
2554#  2554    2556#  2556    2558#  2558    2560#  2560    2562#  2562    2564#  2564    2566#  2566
2568#  2568    2570#  2570    2572#  2572    2574#  2574    2576#  2576    2578#  2578    2580#  2580
2582#  2582    2584#  2584    2586#  2586    2588#  2588    2593#  2593    2595#  2595    2597#  2597
2599#  2599    2601#  2601    2603#  2603    2605#  2605    2607#  2607    2609#  2609    2611#  2611
2613#  2613    2615#  2615    2709#  2709    2711#  2711    2713#  2713    2715#  2715    2717#  2717
2719#  2719    2721#  2721    2723#  2723    2725#  2725    2727#  2727    2729#  2729    2731#  2731
2733#  2733    2735#  2735    2737#  2737    2739#  2739    2741#  2741    2743#  2743    2745#  2745
2747#  2747    2749#  2749    2751#  2751    2753#  2753    2755#  2755    2757#  2757    2759#  2759
2761#  2761    2763#  2763    2765#  2765    2767#  2767    2769#  2769    2771#  2771    2773#  2773
2775#  2775    2777#  2777    2779#  2779    2784#  2784    2786#  2786    2788#  2788    2790#  2790
2792#  2792    2794#  2794    2796#  2796    2798#  2798    2800#  2800    2802#  2802    2804#  2804
2806#  2806    2808#  2808    2810#  2810    2812#  2812    2814#  2814    2816#  2816    2818#  2818
2820#  2820    2822#  2822    2824#  2824    2826#  2826    2828#  2828    2830#  2830    2835#  2835
2837#  2837    2839#  2839    2841#  2841    2843#  2843    2845#  2845    2847#  2847    2849#  2849
2851#  2851    2853#  2853    2855#  2855    2857#  2857    2859#  2859    2861#  2861    2955#  2955
2957#  2957    2959#  2959    2961#  2961    2963#  2963    2965#  2965    2967#  2967    2969#  2969
2971#  2971    2973#  2973    2975#  2975    2977#  2977    2979#  2979    2981#  2981    2983#  2983
2985#  2985    2987#  2987    2989#  2989    2991#  2991    2993#  2993    2995#  2995    2997#  2997
2999#  2999    3001#  3001    3003#  3003    3008#  3008    3010#  3010    3012#  3012    3014#  3014
3016#  3016    3018#  3018    3020#  3020    3022#  3022    3024#  3024    3026#  3026    3028#  3028
3030#  3030    3032#  3032    3034#  3034    3036#  3036    3038#  3038    3040#  3040    3042#  3042
3044#  3044    3046#  3046    3048#  3048    3050#  3050    3052#  3052    3054#  3054    3056#  3056
3058#  3058    3060#  3060    3062#  3062    3064#  3064    3066#  3066    3071#  3071    3073#  3073
3075#  3075    3077#  3077    3079#  3079    3081#  3081    3083#  3083    3085#  3085    3087#  3087
3089#  3089    3091#  3091    3093#  3093    3095#  3095    3097#  3097    3099#  3099    3101#  3101
3103#  3103    3105#  3105    3107#  3107    3109#  3109    3111#  3111    3113#  3113    3115#  3115
3117#  3117    3119#  3119    3121#  3121    3123#  3123    3125#  3125    3127#  3127    3129#  3129
3131#  3131    3133#  3133    3135#  3135    3137#  3137    3139#  3139    3141#  3141    3143#  3143
3145#  3145    3147#  3147    3149#  3149    3243#  3243    3245#  3245    3247#  3247    3250#  3250
3252#  3252    3254#  3254    3256#  3256    3258#  3258    3260#  3260    3262#  3262    3264#  3264
3266#  3266    3268#  3268    3270#  3270    3272#  3272    3274#  3274    3276#  3276    3278#  3278
3280#  3280    3282#  3282    3284#  3284    3286#  3286    3288#  3288    3290#  3290    3292#  3292
3294#  3294    3296#  3296    3298#  3298    3300#  3300    3302#  3302    3304#  3304    3306#  3306
3308#  3308    3310#  3310    3312#  3312    3314#  3314    3316#  3316    3318#  3318    3320#  3320
3325#  3325    3327#  3327    3329#  3329    3331#  3331    3333#  3333    3335#  3335    3337#  3337
3339#  3339    3341#  3341    3343#  3343    3345#  3345    3347#  3347    3349#  3349    3351#  3351
3353#  3353    3355#  3355    3357#  3357    3359#  3359    3361#  3361    3363#  3363    3365#  3365
3367#  3367    3369#  3369    3371#  3371    3373#  3373    3375#  3375    3377#  3377    3379#  3379
3384#  3384    3386#  3386    3388#  3388    3390#  3390    3392#  3392    3394#  3394    3396#  3396
```

```
          3398#  3398    3400#  3400    3402#  3402    3404#  3404    3406#  3406    3408#  3408    3410#  3410      SEQ 1049
          3412#  3412    3414#  3414    3416#  3416    3418#  3418    3420#  3420    3422#  3422    3424#  3424
          3426#  3426    3428#  3428    3430#  3430    3432#  3432    3434#  3434    3436#  3436    3438#  3438
          3440#  3440    3442#  3442    3444#  3444    3446#  3446    3448#  3448    3450#  3450    3452#  3452
          3454#  3454    3456#  3456    3458#  3458    3460#  3460    3462#  3462    3464#  3464    3466#  3466
          3468#  3468    3470#  3470    3472#  3472    3474#  3474    3476#  3476    3478#  3478    3480#  3480
          3482#  3482    3484#  3484    3486#  3486    3488#  3488    3490#  3490    3492#  3492    3494#  3494
          3496#  3496    3498#  3498    3500#  3500    3502#  3502    3504#  3504    3506#  3506    3508#  3508
          3510#  3510    3512#  3512    3514#  3514    3516#  3516    3724#  3724    3726#  3726    3728#  3728
          3730#  3730    3732#  3732    3734#  3734    3736#  3736    3738#  3738    3740#  3740    3742#  3742
          3747#  3747    3749#  3749    3751#  3751    3753#  3753    3755#  3755    3757#  3757    3759#  3759
          3761#  3761    3766#  3766    3768#  3768    3770#  3770    3772#  3772    3777#  3777    3779#  3779
          3781#  3781    3783#  3783    3785#  3785    3787#  3787    3789#  3789    3791#  3791    3796#  3796
          3798#  3798    3800#  3800    3802#  3802    3804#  3804    3806#  3806    3808#  3808    3810#  3810
          3812#  3812    3817#  3817    3819#  3819    3821#  3821    3823#  3823    3825#  3825    3830#  3830
          3832#  3832    3834#  3834    3839#  3839    3841#  3841    3843#  3843    4055#  4055    4057#  4057
          4059#  4059    4061#  4061    4063#  4063    4065#  4065    4067#  4067    4069#  4069    4071#  4071
          4073#  4073    4078#  4078    4080#  4080    4082#  4082    4084#  4084    4086#  4086    4088#  4088
          4090#  4090    4092#  4092    4097#  4097    4099#  4099    4101#  4101    4103#  4103    4108#  4108
          4110#  4110    4112#  4112    4114#  4114    4116#  4116    4118#  4118    4120#  4120    4122#  4122
          4127#  4127    4129#  4129    4131#  4131    4133#  4133    4135#  4135    4137#  4137    4139#  4139
          4141#  4141    4143#  4143    4148#  4148    4150#  4150    4152#  4152    4154#  4154    4156#  4156
          4161#  4161    4163#  4163    4165#  4165    4170#  4170    4172#  4172    4174#  4174    4385#  4385
          4387#  4387    4389#  4389    4391#  4391    4393#  4393    4395#  4395    4397#  4397    4402#  4402
          4404#  4404    4406#  4406    4408#  4408    4410#  4410    4412#  4412    4414#  4414    4419#  4419
          4421#  4421    4423#  4423    4425#  4425    4430#  4430    4432#  4432    4434#  4434    4436#  4436
          4438#  4438    4440#  4440    4442#  4442    4447#  4447    4449#  4449    4451#  4451    4453#  4453
          4455#  4455    4457#  4457    4459#  4459    4461#  4461    4466#  4466    4468#  4468    4470#  4470
          4472#  4472    4474#  4474    4479#  4479    4481#  4481    4483#  4483    4488#  4488    4490#  4490
          4492#  4492    4717#  4717    4719#  4719    4721#  4721    4723#  4723    4725#  4725    4730#  4730
          4732#  4732    4734#  4734    4736#  4736    4738#  4738    4740#  4740    4742#  4742    4747#  4747
          4749#  4749    4751#  4751    4753#  4753    4758#  4758    4760#  4760    4765#  4765    4767#  4767
          4769#  4769    4774#  4774    4776#  4776    4778#  4778    4780#  4780    4782#  4782    4787#  4787
          4789#  4789    4791#  4791    4796#  4796    4798#  4798    4800#  4800    5009#  5009    5011#  5011
          5013#  5013    5015#  5015    5017#  5017    5019#  5019    5021#  5021    5026#  5026    5028#  5028
          5030#  5030    5032#  5032    5034#  5034    5036#  5036    5038#  5038    5043#  5043    5045#  5045
          5047#  5047    5049#  5049    5054#  5054    5056#  5056    5058#  5058    5060#  5060    5062#  5062
          5064#  5064    5066#  5066    5071#  5071    5073#  5073    5075#  5075    5077#  5077    5079#  5079
          5081#  5081    5083#  5083    5085#  5085    5090#  5090    5092#  5092    5094#  5094    5096#  5096
          5098#  5098    5103#  5103    5105#  5105    5107#  5107    5112#  5112    5114#  5114    5116#  5116
          5328#  5328    5330#  5330    5332#  5332    5334#  5334    5336#  5336    5341#  5341    5343#  5343
          5345#  5345    5347#  5347    5349#  5349    5351#  5351    5353#  5353    5358#  5358    5360#  5360
          5362#  5362    5364#  5364    5369#  5369    5371#  5371    5376#  5376    5378#  5378    5380#  5380
          5385#  5385    5387#  5387    5389#  5389    5391#  5391    5393#  5393    5398#  5398    5400#  5400
          5402#  5402    5407#  5407    5409#  5409    5411#  5411
$GARG      198#   198     200#   200     202#   202     204#   204     209#   209     211#   211     213#   213
           215#   215     220#   220     222#   222     224#   224     226#   226     231#   231     233#   233
           235#   235     237#   237     242#   242     244#   244     246#   246     248#   248     253#   253
           255#   255     257#   257     259#   259     264#   264     266#   266     268#   268     270#   270
           275#   275     277#   277     279#   279     281#   281     283#   283     285#   285     287#   287
           391#   391     393#   393     395#   395     397#   397     402#   402     404#   404     406#   406
           408#   408     413#   413     415#   415     417#   417     419#   419     424#   424     426#   426
           428#   428     430#   430     435#   435     437#   437     439#   439     441#   441     446#   446
```

```
   448#    448   450#   450   452#   452   457#   457   459#   459   461#   461   463#   463
   468#    468   470#   470   472#   472   474#   474   476#   476   478#   478   480#   480
   482#    482   484#   484   486#   486   488#   488   490#   490   492#   492   494#   494
   499#    499   501#   501   503#   503   505#   505   510#   510   512#   512   514#   514
   516#    516   521#   521   523#   523   525#   525   527#   527   532#   532   534#   534
   536#    536   538#   538   543#   543   545#   545   547#   547   549#   549   554#   554
   556#    556   558#   558   560#   560   565#   565   567#   567   569#   569   571#   571
   576#    576   578#   578   580#   580   582#   582   584#   584   586#   586   588#   588
   590#    590   592#   592   594#   594   596#   596   598#   598   600#   600   602#   602
   607#    607   609#   609   611#   611   613#   613   615#   615   617#   617   619#   619
   621#    621   623#   623   625#   625   627#   627   629#   629   631#   631   633#   633
   739#    739   741#   741   743#   743   745#   745   829#   829   831#   831   833#   833
   835#    835   919#   919   921#   921   923#   923   925#   925  1018#  1018  1020#  1020
  1022#   1022  1024#  1024  1117#  1117  1119#  1119  1121#  1121  1123#  1123  1216#  1216
  1218#   1218  1220#  1220  1222#  1222  1315#  1315  1317#  1317  1319#  1319  1321#  1321
  1416#   1416  1418#  1418  1420#  1420  1422#  1422  1577#  1577  1579#  1579  1581#  1581
  1584#   1584  1586#  1586  1588#  1588  1593#  1593  1595#  1595  1597#  1597  1600#  1600
  1602#   1602  1604#  1604  1609#  1609  1611#  1611  1613#  1613  1616#  1616  1618#  1618
  1620#   1620  1625#  1625  1627#  1627  1629#  1629  1632#  1632  1634#  1634  1636#  1636
  1641#   1641  1643#  1643  1645#  1645  1648#  1648  1650#  1650  1652#  1652  1657#  1657
  1659#   1659  1661#  1661  1664#  1664  1666#  1666  1668#  1668  1673#  1673  1675#  1675
  1677#   1677  1680#  1680  1682#  1682  1684#  1684  1689#  1689  1691#  1691  1693#  1693
  1696#   1696  1698#  1698  1700#  1700  1705#  1705  1707#  1707  1709#  1709  1712#  1712
  1714#   1714  1716#  1716  1721#  1721  1723#  1723  1725#  1725  1728#  1728  1730#  1730
  1732#   1732  1737#  1737  1739#  1739  1741#  1741  1744#  1744  1746#  1746  1748#  1748
  1753#   1753  1755#  1755  1757#  1757  1760#  1760  1762#  1762  1764#  1764  1769#  1769
  1771#   1771  1773#  1773  1776#  1776  1778#  1778  1780#  1780  1785#  1785  1787#  1787
  1789#   1789  1792#  1792  1794#  1794  1796#  1796  1801#  1801  1803#  1803  1805#  1805
  1808#   1808  1810#  1810  1812#  1812  1817#  1817  1819#  1819  1821#  1821  1824#  1824
  1826#   1826  1828#  1828  1912#  1912  1914#  1914  1916#  1916  1919#  1919  1921#  1921
  1923#   1923  1925#  1925  2051#  2051  2053#  2053  2055#  2055  2057#  2057  2059#  2059
  2061#   2061  2063#  2063  2065#  2065  2067#  2067  2069#  2069  2071#  2071  2073#  2073
  2075#   2075  2077#  2077  2079#  2079  2081#  2081  2083#  2083  2088#  2088  2090#  2090
  2092#   2092  2094#  2094  2096#  2096  2098#  2098  2100#  2100  2102#  2102  2104#  2104
  2106#   2106  2108#  2108  2110#  2110  2112#  2112  2114#  2114  2116#  2116  2118#  2118
  2120#   2120  2245#  2245  2247#  2247  2249#  2249  2251#  2251  2253#  2253  2255#  2255
  2257#   2257  2259#  2259  2261#  2261  2263#  2263  2265#  2265  2267#  2267  2269#  2269
  2271#   2271  2273#  2273  2275#  2275  2277#  2277  2279#  2279  2281#  2281  2283#  2283
  2285#   2285  2287#  2287  2289#  2289  2291#  2291  2293#  2293  2295#  2295  2297#  2297
  2299#   2299  2301#  2301  2303#  2303  2305#  2305  2307#  2307  2309#  2309  2311#  2311
  2313#   2313  2318#  2318  2320#  2320  2322#  2322  2324#  2324  2326#  2326  2328#  2328
  2330#   2330  2332#  2332  2334#  2334  2336#  2336  2338#  2338  2340#  2340  2342#  2342
  2344#   2344  2346#  2346  2348#  2348  2350#  2350  2352#  2352  2357#  2357  2359#  2359
  2361#   2361  2363#  2363  2365#  2365  2367#  2367  2369#  2369  2371#  2371  2373#  2373
  2375#   2375  2377#  2377  2379#  2379  2381#  2381  2475#  2475  2477#  2477  2479#  2479
  2481#   2481  2483#  2483  2485#  2485  2487#  2487  2489#  2489  2491#  2491  2493#  2493
  2495#   2495  2497#  2497  2499#  2499  2501#  2501  2503#  2503  2505#  2505  2507#  2507
  2509#   2509  2511#  2511  2513#  2513  2515#  2515  2517#  2517  2519#  2519  2521#  2521
  2523#   2523  2525#  2525  2527#  2527  2529#  2529  2531#  2531  2533#  2533  2535#  2535
  2537#   2537  2539#  2539  2541#  2541  2543#  2543  2545#  2545  2550#  2550  2552#  2552
  2554#   2554  2556#  2556  2558#  2558  2560#  2560  2562#  2562  2564#  2564  2566#  2566
  2568#   2568  2570#  2570  2572#  2572  2574#  2574  2576#  2576  2578#  2578  2580#  2580
  2582#   2582  2584#  2584  2586#  2586  2588#  2588  2593#  2593  2595#  2595  2597#  2597
```

```
2599#  2599    2601#  2601    2603#  2603    2605#  2605    2607#  2607    2609#  2609    2611#  2611      SEQ 1051
2613#  2613    2615#  2615    2709#  2709    2711#  2711    2713#  2713    2715#  2715    2717#  2717
2719#  2719    2721#  2721    2723#  2723    2725#  2725    2727#  2727    2729#  2729    2731#  2731
2733#  2733    2735#  2735    2737#  2737    2739#  2739    2741#  2741    2743#  2743    2745#  2745
2747#  2747    2749#  2749    2751#  2751    2753#  2753    2755#  2755    2757#  2757    2759#  2759
2761#  2761    2763#  2763    2765#  2765    2767#  2767    2769#  2769    2771#  2771    2773#  2773
2775#  2775    2777#  2777    2779#  2779    2784#  2784    2786#  2786    2788#  2788    2790#  2790
2792#  2792    2794#  2794    2796#  2796    2798#  2798    2800#  2800    2802#  2802    2804#  2804
2806#  2806    2808#  2808    2810#  2810    2812#  2812    2814#  2814    2816#  2816    2818#  2818
2820#  2820    2822#  2822    2824#  2824    2826#  2826    2828#  2828    2830#  2830    2835#  2835
2837#  2837    2839#  2839    2841#  2841    2843#  2843    2845#  2845    2847#  2847    2849#  2849
2851#  2851    2853#  2853    2855#  2855    2857#  2857    2859#  2859    2861#  2861    2955#  2955
2957#  2957    2959#  2959    2961#  2961    2963#  2963    2965#  2965    2967#  2967    2969#  2969
2971#  2971    2973#  2973    2975#  2975    2977#  2977    2979#  2979    2981#  2981    2983#  2983
2985#  2985    2987#  2987    2989#  2989    2991#  2991    2993#  2993    2995#  2995    2997#  2997
2999#  2999    3001#  3001    3003#  3003    3008#  3008    3010#  3010    3012#  3012    3014#  3014
3016#  3016    3018#  3018    3020#  3020    3022#  3022    3024#  3024    3026#  3026    3028#  3028
3030#  3030    3032#  3032    3034#  3034    3036#  3036    3038#  3038    3040#  3040    3042#  3042
3044#  3044    3046#  3046    3048#  3048    3050#  3050    3052#  3052    3054#  3054    3056#  3056
3058#  3058    3060#  3060    3062#  3062    3064#  3064    3066#  3066    3071#  3071    3073#  3073
3075#  3075    3077#  3077    3079#  3079    3081#  3081    3083#  3083    3085#  3085    3087#  3087
3089#  3089    3091#  3091    3093#  3093    3095#  3095    3097#  3097    3099#  3099    3101#  3101
3103#  3103    3105#  3105    3107#  3107    3109#  3109    3111#  3111    3113#  3113    3115#  3115
3117#  3117    3119#  3119    3121#  3121    3123#  3123    3125#  3125    3127#  3127    3129#  3129
3131#  3131    3133#  3133    3135#  3135    3137#  3137    3139#  3139    3141#  3141    3143#  3143
3145#  3145    3147#  3147    3149#  3149    3243#  3243    3245#  3245    3247#  3247    3250#  3250
3252#  3252    3254#  3254    3256#  3256    3258#  3258    3260#  3260    3262#  3262    3264#  3264
3266#  3266    3268#  3268    3270#  3270    3272#  3272    3274#  3274    3276#  3276    3278#  3278
3280#  3280    3282#  3282    3284#  3284    3286#  3286    3288#  3288    3290#  3290    3292#  3292
3294#  3294    3296#  3296    3298#  3298    3300#  3300    3302#  3302    3304#  3304    3306#  3306
3308#  3308    3310#  3310    3312#  3312    3314#  3314    3316#  3316    3318#  3318    3320#  3320
3325#  3325    3327#  3327    3329#  3329    3331#  3331    3333#  3333    3335#  3335    3337#  3337
3339#  3339    3341#  3341    3343#  3343    3345#  3345    3347#  3347    3349#  3349    3351#  3351
3353#  3353    3355#  3355    3357#  3357    3359#  3359    3361#  3361    3363#  3363    3365#  3365
3367#  3367    3369#  3369    3371#  3371    3373#  3373    3375#  3375    3377#  3377    3379#  3379
3384#  3384    3386#  3386    3388#  3388    3390#  3390    3392#  3392    3394#  3394    3396#  3396
3398#  3398    3400#  3400    3402#  3402    3404#  3404    3406#  3406    3408#  3408    3410#  3410
3412#  3412    3414#  3414    3416#  3416    3418#  3418    3420#  3420    3422#  3422    3424#  3424
3426#  3426    3428#  3428    3430#  3430    3432#  3432    3434#  3434    3436#  3436    3438#  3438
3440#  3440    3442#  3442    3444#  3444    3446#  3446    3448#  3448    3450#  3450    3452#  3452
3454#  3454    3456#  3456    3458#  3458    3460#  3460    3462#  3462    3464#  3464    3466#  3466
3468#  3468    3470#  3470    3472#  3472    3474#  3474    3476#  3476    3478#  3478    3480#  3480
3482#  3482    3484#  3484    3486#  3486    3488#  3488    3490#  3490    3492#  3492    3494#  3494
3496#  3496    3498#  3498    3500#  3500    3502#  3502    3504#  3504    3506#  3506    3508#  3508
3510#  3510    3512#  3512    3514#  3514    3516#  3516    3724#  3724    3726#  3726    3728#  3728
3730#  3730    3732#  3732    3734#  3734    3736#  3736    3738#  3738    3740#  3740    3742#  3742
3747#  3747    3749#  3749    3751#  3751    3753#  3753    3755#  3755    3757#  3757    3759#  3759
3761#  3761    3766#  3766    3768#  3768    3770#  3770    3772#  3772    3777#  3777    3779#  3779
3781#  3781    3783#  3783    3785#  3785    3787#  3787    3789#  3789    3791#  3791    3796#  3796
3798#  3798    3800#  3800    3802#  3802    3804#  3804    3806#  3806    3808#  3808    3810#  3810
3812#  3812    3817#  3817    3819#  3819    3821#  3821    3823#  3823    3825#  3825    3830#  3830
3832#  3832    3834#  3834    3839#  3839    3841#  3841    3843#  3843    4055#  4055    4057#  4057
4059#  4059    4061#  4061    4063#  4063    4065#  4065    4067#  4067    4069#  4069    4071#  4071
```

```
          4073#  4073   4078#  4078   4080#  4080   4082#  4082   4084#  4084   4086#  4086   4088#  4088
          4090#  4090   4092#  4092   4097#  4097   4099#  4099   4101#  4101   4103#  4103   4108#  4108
          4110#  4110   4112#  4112   4114#  4114   4116#  4116   4118#  4118   4120#  4120   4122#  4122
          4127#  4127   4129#  4129   4131#  4131   4133#  4133   4135#  4135   4137#  4137   4139#  4139
          4141#  4141   4143#  4143   4148#  4148   4150#  4150   4152#  4152   4154#  4154   4156#  4156
          4161#  4161   4163#  4163   4165#  4165   4170#  4170   4172#  4172   4174#  4174   4385#  4385
          4387#  4387   4389#  4389   4391#  4391   4393#  4393   4395#  4395   4397#  4397   4402#  4402
          4404#  4404   4406#  4406   4408#  4408   4410#  4410   4412#  4412   4414#  4414   4419#  4419
          4421#  4421   4423#  4423   4425#  4425   4430#  4430   4432#  4432   4434#  4434   4436#  4436
          4438#  4438   4440#  4440   4442#  4442   4447#  4447   4449#  4449   4451#  4451   4453#  4453
          4455#  4455   4457#  4457   4459#  4459   4461#  4461   4466#  4466   4468#  4468   4470#  4470
          4472#  4472   4474#  4474   4479#  4479   4481#  4481   4483#  4483   4488#  4488   4490#  4490
          4492#  4492   4717#  4717   4719#  4719   4721#  4721   4723#  4723   4725#  4725   4730#  4730
          4732#  4732   4734#  4734   4736#  4736   4738#  4738   4740#  4740   4742#  4742   4747#  4747
          4749#  4749   4751#  4751   4753#  4753   4758#  4758   4760#  4760   4765#  4765   4767#  4767
          4769#  4769   4774#  4774   4776#  4776   4778#  4778   4780#  4780   4782#  4782   4787#  4787
          4789#  4789   4791#  4791   4796#  4796   4798#  4798   4800#  4800   5009#  5009   5011#  5011
          5013#  5013   5015#  5015   5017#  5017   5019#  5019   5021#  5021   5026#  5026   5028#  5028
          5030#  5030   5032#  5032   5034#  5034   5036#  5036   5038#  5038   5043#  5043   5045#  5045
          5047#  5047   5049#  5049   5054#  5054   5056#  5056   5058#  5058   5060#  5060   5062#  5062
          5064#  5064   5066#  5066   5071#  5071   5073#  5073   5075#  5075   5077#  5077   5079#  5079
          5081#  5081   5083#  5083   5085#  5085   5090#  5090   5092#  5092   5094#  5094   5096#  5096
          5098#  5098   5103#  5103   5105#  5105   5107#  5107   5112#  5112   5114#  5114   5116#  5116
          5328#  5328   5330#  5330   5332#  5332   5334#  5334   5336#  5336   5341#  5341   5343#  5343
          5345#  5345   5347#  5347   5349#  5349   5351#  5351   5353#  5353   5358#  5358   5360#  5360
          5362#  5362   5364#  5364   5369#  5369   5371#  5371   5376#  5376   5378#  5378   5380#  5380
          5385#  5385   5387#  5387   5389#  5389   5391#  5391   5393#  5393   5398#  5398   5400#  5400
          5402#  5402   5407#  5407   5409#  5409   5411#  5411
%ADDR      198#   198    200    200#   202    202#   204    204#   209    209#   211    211#   213    213#
           215    215#   220    220#   222    222#   224    224#   226    226#   231    231#   233    233#
           235    235#   237    237#   242    242#   244    244#   246    246#   248    248#   253    253#
           255    255#   257    257#   259    259#   264    264#   266    266#   268    268#   270    270#
           275    275#   277    277#   279    279#   281    281#   283    283#   285    285#   287    287#
           391#   [illegible]   393    393#   395    395#   397    397#   402    402#   404    404#   406    406#
           408    408#   413    413#   415    415#   417    417#   419    419#   424    424#   426    426#
           428    428#   430    430#   435    435#   437    437#   439    439#   441    441#   446    446#
           448    448#   450    450#   452    452#   457    457#   459    459#   461    461#   463    463#
           468    468#   470    470#   472    472#   474    474#   476    476#   478    478#   480    480#
           [illegible]   482#   484    484#   486    486#   488    488#   490    490#   492    492#   494    494#
           [illegible]   499#   501    501#   503    503#   505    505#   510    510#   512    512#   514    514#
           [illegible]   516#   521    521#   523    523#   525    525#   527    527#   532    532#   534    534#
           536    [illegible]   538    538#   543    543#   545    545#   547    547#   549    549#   554    554#
           556    [illegible]   558    558#   560    560#   565    565#   567    567#   569    569#   571    571#
           576    [illegible]   578    578#   580    580#   582    582#   584    584#   586    586#   588    588#
           [illegible]   [illegible]   592    592#   594    594#   596    596#   598    598#   600    600#   602    602#
           [illegible]   607#   609    609#   611    611#   613    613#   615    615#   617    617#   619    619#
           [illegible]   621#   623    623#   625    625#   627    627#   629    629#   631    631#   633    633#
           739#   739    741    741#   743    743#   745    745#   829#   829    831    831#   833    833#
           835    835#   919#   919    921    921#   923    923#   925    925#  1018#  1018   1020   1020#
          1022   1022#  1024   1024#  1117#  1117   1119   1119#  1121   1121#  1123   1123#  1216#  1216
          1218   1218#  1220   1220#  1222   1222#  1315#  1315   1317   1317#  1319   1319#  1321   1321#
          1416#  1416   1418   1418#  1420   1420#  1422   1422#  1577#  1577   1579   1579#  1581   1581#
          1584   1584#  1586   1586#  1588   1588#  1593   1593#  1595   1595#  1597   1597#  1600   1600#
```

```
1602    1602#   1604    1604#   1609    1609#   1611    1611#   1613    1613#   1616    1616#   1618    1618#     SEQ 1053
1620    1620#   1625    1625#   1627    1627#   1629    1629#   1632    1632#   1634    1634#   1636    1636#
1641    1641#   1643    1643#   1645    1645#   1648    1648#   1650    1650#   1652    1652#   1657    1657#
1659    1659#   1661    1661#   1664    1664#   1666    1666#   1668    1668#   1673    1673#   1675    1675#
1677    1677#   1680    1680#   1682    1682#   1684    1684#   1689    1689#   1691    1691#   1693    1693#
1696    1696#   1698    1698#   1700    1700#   1705    1705#   1707    1707#   1709    1709#   1712    1712#
1714    1714#   1716    1716#   1721    1721#   1723    1723#   1725    1725#   1728    1728#   1730    1730#
1732    1732#   1737    1737#   1739    1739#   1741    1741#   1744    1744#   1746    1746#   1748    1748#
1753    1753#   1755    1755#   1757    1757#   1760    1760#   1762    1762#   1764    1764#   1769    1769#
1771    1771#   1773    1773#   1776    1776#   1778    1778#   1780    1780#   1785    1785#   1787    1787#
1789    1789#   1792    1792#   1794    1794#   1796    1796#   1801    1801#   1803    1803#   1805    1805#
1808    1808#   1810    1810#   1812    1812#   1817    1817#   1819    1819#   1821    1821#   1824    1824#
1826    1826#   1828    1828#   1912#   1912    1914    1914#   1916    1916#   1919    1919#   1921    1921#
1923    1923#   1925    1925#   2051#   2051    2053    2053#   2055    2055#   2057    2057#   2059    2059#
2061    2061#   2063    2063#   2065    2065#   2067    2067#   2069    2069#   2071    2071#   2073    2073#
2075    2075#   2077    2077#   2079    2079#   2081    2081#   2083    2083#   2088    2088#   2090    2090#
2092    2092#   2094    2094#   2096    2096#   2098    2098#   2100    2100#   2102    2102#   2104    2104#
2106    2106#   2108    2108#   2110    2110#   2112    2112#   2114    2114#   2116    2116#   2118    2118#
2120    2120#   2245#   2245    2247    2247#   2249    2249#   2251    2251#   2253    2253#   2255    2255#
2257    2257#   2259    2259#   2261    2261#   2263    2263#   2265    2265#   2267    2267#   2269    2269#
2271    2271#   2273    2273#   2275    2275#   2277    2277#   2279    2279#   2281    2281#   2283    2283#
2285    2285#   2287    2287#   2289    2289#   2291    2291#   2293    2293#   2295    2295#   2297    2297#
2299    2299#   2301    2301#   2303    2303#   2305    2305#   2307    2307#   2309    2309#   2311    2311#
2313    2313#   2318    2318#   2320    2320#   2322    2322#   2324    2324#   2326    2326#   2328    2328#
2330    2330#   2332    2332#   2334    2334#   2336    2336#   2338    2338#   2340    2340#   2342    2342#
2344    2344#   2346    2346#   2348    2348#   2350    2350#   2352    2352#   2357    2357#   2359    2359#
2361    2361#   2363    2363#   2365    2365#   2367    2367#   2369    2369#   2371    2371#   2373    2373#
2375    2375#   2377    2377#   2379    2379#   2381    2381#   2475#   2475    2477    2477#   2479    2479#
2481    2481#   2483    2483#   2485    2485#   2487    2487#   2489    2489#   2491    2491#   2493    2493#
2495    2495#   2497    2497#   2499    2499#   2501    2501#   2503    2503#   2505    2505#   2507    2507#
2509    2509#   2511    2511#   2513    2513#   2515    2515#   2517    2517#   2519    2519#   2521    2521#
2523    2523#   2525    2525#   2527    2527#   2529    2529#   2531    2531#   2533    2533#   2535    2535#
2537    2537#   2539    2539#   2541    2541#   2543    2543#   2545    2545#   2550    2550#   2552    2552#
2554    2554#   2556    2556#   2558    2558#   2560    2560#   2562    2562#   2564    2564#   2566    2566#
2568    2568#   2570    2570#   2572    2572#   2574    2574#   2576    2576#   2578    2578#   2580    2580#
2582    2582#   2584    2584#   2586    2586#   2588    2588#   2593    2593#   2595    2595#   2597    2597#
2599    2599#   2601    2601#   2603    2603#   2605    2605#   2607    2607#   2609    2609#   2611    2611#
2613    2613#   2615    2615#   2709#   2709    2711    2711#   2713    2713#   2715    2715#   2717    2717#
2719    2719#   2721    2721#   2723    2723#   2725    2725#   2727    2727#   2729    2729#   2731    2731#
2733    2733#   2735    2735#   2737    2737#   2739    2739#   2741    2741#   2743    2743#   2745    2745#
2747    2747#   2749    2749#   2751    2751#   2753    2753#   2755    2755#   2757    2757#   2759    2759#
2761    2761#   2763    2763#   2765    2765#   2767    2767#   2769    2769#   2771    2771#   2773    2773#
2775    2775#   2777    2777#   2779    2779#   2784    2784#   2786    2786#   2788    2788#   2790    2790#
2792    2792#   2794    2794#   2796    2796#   2798    2798#   2800    2800#   2802    2802#   2804    2804#
2806    2806#   2808    2808#   2810    2810#   2812    2812#   2814    2814#   2816    2816#   2818    2818#
2820    2820#   2822    2822#   2824    2824#   2826    2826#   2828    2828#   2830    2830#   2835    2835#
2837    2837#   2839    2839#   2841    2841#   2843    2843#   2845    2845#   2847    2847#   2849    2849#
2851    2851#   2853    2853#   2855    2855#   2857    2857#   2859    2859#   2861    2861#   2955#   2955
2957    2957#   2959    2959#   2961    2961#   2963    2963#   2965    2965#   2967    2967#   2969    2969#
2971    2971#   2973    2973#   2975    2975#   2977    2977#   2979    2979#   2981    2981#   2983    2983#
2985    2985#   2987    2987#   2989    2989#   2991    2991#   2993    2993#   2995    2995#   2997    2997#
2999    2999#   3001    3001#   3003    3003#   3008    3008#   3010    3010#   3012    3012#   3014    3014#
3016    3016#   3018    3018#   3020    3020#   3022    3022#   3024    3024#   3026    3026#   3028    3028#
```

```
3030     3030#    3032     3032#    3034     3034#    3036     3036#    3038     3038#    3040     3040#    3042     3042#
3044     3044#    3046     3046#    3048     3048#    3050     3050#    3052     3052#    3054     3054#    3056     3056#        SEQ 1054
3058     3058#    3060     3060#    3062     3062#    3064     3064#    3066     3066#    3071     3071#    3073     3073#
3075     3075#    3077     3077#    3079     3079#    3081     3081#    3083     3083#    3085     3085#    3087     3087#
3089     3089#    3091     3091#    3093     3093#    3095     3095#    3097     3097#    3099     3099#    3101     3101#
3103     3103#    3105     3105#    3107     3107#    3109     3109#    3111     3111#    3113     3113#    3115     3115#
3117     3117#    3119     3119#    3121     3121#    3123     3123#    3125     3125#    3127     3127#    3129     3129#
3131     3131#    3133     3133#    3135     3135#    3137     3137#    3139     3139#    3141     3141#    3143     3143#
3145     3145#    3147     3147#    3149     3149#    3243#    3243     3245     3245#    3247     3247#    3250     3250#
3252     3252#    3254     3254#    3256     3256#    3258     3258#    3260     3260#    3262     3262#    3264     3264#
3266     3266#    3268     3268#    3270     3270#    3272     3272#    3274     3274#    3276     3276#    3278     3278#
3280     3280#    3282     3282#    3284     3284#    3286     3286#    3288     3288#    3290     3290#    3292     3292#
3294     3294#    3296     3296#    3298     3298#    3300     3300#    3302     3302#    3304     3304#    3306     3306#
3308     3308#    3310     3310#    3312     3312#    3314     3314#    3316     3316#    3318     3318#    3320     3320#
3325     3325#    3327     3327#    3329     3329#    3331     3331#    3333     3333#    3335     3335#    3337     3337#
3339     3339#    3341     3341#    3343     3343#    3345     3345#    3347     3347#    3349     3349#    3351     3351#
3353     3353#    3355     3355#    3357     3357#    3359     3359#    3361     3361#    3363     3363#    3365     3365#
3367     3367#    3369     3369#    3371     3371#    3373     3373#    3375     3375#    3377     3377#    3379     3379#
3384     3384#    3386     3386#    3388     3388#    3390     3390#    3392     3392#    3394     3394#    3396     3396#
3398     3398#    3400     3400#    3402     3402#    3404     3404#    3406     3406#    3408     3408#    3410     3410#
3412     3412#    3414     3414#    3416     3416#    3418     3418#    3420     3420#    3422     3422#    3424     3424#
3426     3426#    3428     3428#    3430     3430#    3432     3432#    3434     3434#    3436     3436#    3438     3438#
3440     3440#    3442     3442#    3444     3444#    3446     3446#    3448     3448#    3450     3450#    3452     3452#
3454     3454#    3456     3456#    3458     3458#    3460     3460#    3462     3462#    3464     3464#    3466     3466#
3468     3468#    3470     3470#    3472     3472#    3474     3474#    3476     3476#    3478     3478#    3480     3480#
3482     3482#    3484     3484#    3486     3486#    3488     3488#    3490     3490#    3492     3492#    3494     3494#
3496     3496#    3498     3498#    3500     3500#    3502     3502#    3504     3504#    3506     3506#    3508     3508#
3510     3510#    3512     3512#    3514     3514#    3516     3516#    3724#    3724     3726     3726#    3728     3728#
3730     3730#    3732     3732#    3734     3734#    3736     3736#    3738     3738#    3740     3740#    3742     3742#
3747     3747#    3749     3749#    3751     3751#    3753     3753#    3755     3755#    3757     3757#    3759     3759#
3761     3761#    3766     3766#    3768     3768#    3770     3770#    3772     3772#    3777     3777#    3779     3779#
3781     3781#    3783     3783#    3785     3785#    3787     3787#    3789     3789#    3791     3791#    3796     3796#
3798     3798#    3800     3800#    3802     3802#    3804     3804#    3806     3806#    3808     3808#    3810     3810#
3812     3812#    3817     3817#    3819     3819#    3821     3821#    3823     3823#    3825     3825#    3830#    3830
3832     3832#    3834     3834#    3839     3839#    3841     3841#    3843     3843#    4055#    4055     4057     4057#
4059     4059#    4061     4061#    4063     4063#    4065     4065#    4067     4067#    4069     4069#    4071     4071#
4073     4073#    4078     4078#    4080     4080#    4082     4082#    4084     4084#    4086     4086#    4088     4088#
4090     4090#    4092     4092#    4097     4097#    4099     4099#    4101     4101#    4103     4103#    4108     4108#
4110     4110#    4112     4112#    4114     4114#    4116     4116#    4118     4118#    4120     4120#    4122     4122#
4127     4127#    4129     4129#    4131     4131#    4133     4133#    4135     4135#    4137     4137#    4139     4139#
4141     4141#    4143     4143#    4148     4148#    4150     4150#    4152     4152#    4154     4154#    4156     4156#
4161#    4161     4163     4163#    4165     4165#    4170     4170#    4172     4172#    4174     4174#    4385#    4385
4387     4387#    4389     4389#    4391     4391#    4393     4393#    4395     4395#    4397     4397#    4402     4402#
4404     4404#    4406     4406#    4408     4408#    4410     4410#    4412     4412#    4414     4414#    4419     4419#
4421     4421#    4423     4423#    4425     4425#    4430     4430#    4432     4432#    4434     4434#    4436     4436#
4438     4438#    4440     4440#    4442     4442#    4447     4447#    4449     4449#    4451     4451#    4453     4453#
4455     4455#    4457     4457#    4459     4459#    4461     4461#    4466     4466#    4468     4468#    4470     4470#
4472     4472#    4474     4474#    4479#    4479     4481     4481#    4483     4483#    4488     4488#    4490     4490#
4492     4492#    4717#    4717     4719     4719#    4721     4721#    4723     4723#    4725     4725#    4730     4730#
4732     4732#    4734     4734#    4736     4736#    4738     4738#    4740     4740#    4742     4742#    4747#    4747
4749     4749#    4751     4751#    4753     4753#    4758     4758#    4760     4760#    4765     4765#    4767     4767#
4769     4769#    4774#    4774     4776     4776#    4778     4778#    4780     4780#    4782     4782#    4787#    4787
4789     4789#    4791     4791#    4796     4796#    4798     4798#    4800     4800#    5009#    5009     5011     5011#
```

```
            5013    5013#   5015    5015#   5017    5017#   5019    5019#   5021    5021#   5026    5026#   5028    5028#      SEQ 1055
            5030    5030#   5032    5032#   5034    5034#   5036    5036#   5038    5038#   5043    5043#   5045    5045#
            5047    5047#   5049    5049#   5054    5054#   5056    5056#   5058    5058#   5060    5060#   5062    5062#
            5064    5064#   5066    5066#   5071    5071#   5073    5073#   5075    5075#   5077    5077#   5079    5079#
            5081    5081#   5083    5083#   5085    5085#   5090    5090#   5092    5092#   5094    5094#   5096    5096#
            5098    5098#   5103#   5103    5105    5105#   5107    5107#   5112    5112#   5114    5114#   5116    5116#
            5328#   5328    5330    5330#   5332    5332#   5334    5334#   5336    5336#   5341    5341#   5343    5343#
            5345    5345#   5347    5347#   5349    5349#   5351    5351#   5353    5353#   5358#   5358    5360    5360#
            5362    5362#   5364    5364#   5369    5369#   5371    5371#   5376    5376#   5378    5378#   5380    5380#
            5385#   5385    5387    5387#   5389    5389#   5391    5391#   5393    5393#   5398#   5398    5400    5400#
            5402    5402#   5407    5407#   5409    5409#   5411    5411#
%ML          198#    198     200#    200     202#    202     204#    204     209#    209     211#    211     213#    213
             215#    215     220#    220     222#    222     224#    224     226#    226     231#    231     233#    233
             235#    235     237#    237     242#    242     244#    244     246#    246     248#    248     253#    253
             255#    255     257#    257     259#    259     264#    264     266#    266     268#    268     270#    270
             275#    275     277#    277     279#    279     281#    281     283#    283     285#    285     287#    287
             391#    391     393#    393     395#    395     397#    397     402#    402     404#    404     406#    406
             408#    408     413#    413     415#    415     417#    417     419#    419     424#    424     426#    426
             428#    428     430#    430     435#    435     437#    437     439#    439     441#    441     446#    446
             448#    448     450#    450     452#    452     457#    457     459#    459     461#    461     463#    463
             468#    468     470#    470     472#    472     474#    474     476#    476     478#    478     480#    480
             482#    482     484#    484     486#    486     488#    488     490#    490     492#    492     494#    494
             499#    499     501#    501     503#    503     505#    505     510#    510     512#    512     514#    514
             516#    516     521#    521     523#    523     525#    525     527#    527     532#    532     534#    534
             536#    536     538#    538     543#    543     545#    545     547#    547     549#    549     554#    554
             556#    556     558#    558     560#    560     565#    565     567#    567     569#    569     571#    571
             576#    576     578#    578     580#    580     582#    582     584#    584     586#    586     588#    588
             590#    590     592#    592     594#    594     596#    596     598#    598     600#    600     602#    602
             607#    607     609#    609     611#    611     613#    613     615#    615     617#    617     619#    619
             621#    621     623#    623     625#    625     627#    627     629#    629     631#    631     633#    633
             739#    739     741#    741     743#    743     745#    745     829#    829     831#    831     833#    833
             835#    835     919#    919     921#    921     923#    923     925#    925    1018#   1018    1020#   1020
            1022#   1022    1024#   1024    1117#   1117    1119#   1119    1121#   1121    1123#   1123    1216#   1216
            1218#   1218    1220#   1220    1222#   1222    1315#   1315    1317#   1317    1319#   1319    1321#   1321
            1416#   1416    1418#   1418    1420#   1420    1422#   1422    1577#   1577    1579#   1579    1581#   1581
            1584#   1584    1586#   1586    1588#   1588    1593#   1593    1595#   1595    1597#   1597    1600#   1600
            1602#   1602    1604#   1604    1609#   1609    1611#   1611    1613#   1613    1616#   1616    1618#   1618
            1620#   1620    1625#   1625    1627#   1627    1629#   1629    1632#   1632    1634#   1634    1636#   1636
            1641#   1641    1643#   1643    1645#   1645    1648#   1648    1650#   1650    1652#   1652    1657#   1657
            1659#   1659    1661#   1661    1664#   1664    1666#   1666    1668#   1668    1673#   1673    1675#   1675
            1677#   1677    1680#   1680    1682#   1682    1684#   1684    1689#   1689    1691#   1691    1693#   1693
            1696#   1696    1698#   1698    1700#   1700    1705#   1705    1707#   1707    1709#   1709    1712#   1712
            1714#   1714    1716#   1716    1721#   1721    1723#   1723    1725#   1725    1728#   1728    1730#   1730
            1732#   1732    1737#   1737    1739#   1739    1741#   1741    1744#   1744    1746#   1746    1748#   1748
            1753#   1753    1755#   1755    1757#   1757    1760#   1760    1762#   1762    1764#   1764    1769#   1769
            1771#   1771    1773#   1773    1776#   1776    1778#   1778    1780#   1780    1785#   1785    1787#   1787
            1789#   1789    1792#   1792    1794#   1794    1796#   1796    1801#   1801    1803#   1803    1805#   1805
            1808#   1808    1810#   1810    1812#   1812    1817#   1817    1819#   1819    1821#   1821    1824#   1824
            1826#   1826    1828#   1828    1912#   1912    1914#   1914    1916#   1916    1919#   1919    1921#   1921
            1923#   1923    1925#   1925    2051#   2051    2053#   2053    2055#   2055    2057#   2057    2059#   2059
            2061#   2061    2063#   2063    2065#   2065    2067#   2067    2069#   2069    2071#   2071    2073#   2073
            2075#   2075    2077#   2077    2079#   2079    2081#   2081    2083#   2083    2088#   2088    2090#   2090
            2092#   2092    2094#   2094    2096#   2096    2098#   2098    2100#   2100    2102#   2102    2104#   2104
```

```
2106#  2106    2108#  2108    2110#  2110    2112#  2112    2114#  2114    2116#  2116    2118#  2118      SEQ 1056
2120#  2120    2245#  2245    2247#  2247    2249#  2249    2251#  2251    2253#  2253    2255#  2255
2257#  2257    2259#  2259    2261#  2261    2263#  2263    2265#  2265    2267#  2267    2269#  2269
2271#  2271    2273#  2273    2275#  2275    2277#  2277    2279#  2279    2281#  2281    2283#  2283
2285#  2285    2287#  2287    2289#  2289    2291#  2291    2293#  2293    2295#  2295    2297#  2297
2299#  2299    2301#  2301    2303#  2303    2305#  2305    2307#  2307    2309#  2309    2311#  2311
2313#  2313    2318#  2318    2320#  2320    2322#  2322    2324#  2324    2326#  2326    2328#  2328
2330#  2330    2332#  2332    2334#  2334    2336#  2336    2338#  2338    2340#  2340    2342#  2342
2344#  2344    2346#  2346    2348#  2348    2350#  2350    2352#  2352    2357#  2357    2359#  2359
2361#  2361    2363#  2363    2365#  2365    2367#  2367    2369#  2369    2371#  2371    2373#  2373
2375#  2375    2377#  2377    2379#  2379    2381#  2381    2475#  2475    2477#  2477    2479#  2479
2481#  2481    2483#  2483    2485#  2485    2487#  2487    2489#  2489    2491#  2491    2493#  2493
2495#  2495    2497#  2497    2499#  2499    2501#  2501    2503#  2503    2505#  2505    2507#  2507
2509#  2509    2511#  2511    2513#  2513    2515#  2515    2517#  2517    2519#  2519    2521#  2521
2523#  2523    2525#  2525    2527#  2527    2529#  2529    2531#  2531    2533#  2533    2535#  2535
2537#  2537    2539#  2539    2541#  2541    2543#  2543    2545#  2545    2550#  2550    2552#  2552
2554#  2554    2556#  2556    2558#  2558    2560#  2560    2562#  2562    2564#  2564    2566#  2566
2568#  2568    2570#  2570    2572#  2572    2574#  2574    2576#  2576    2578#  2578    2580#  2580
2582#  2582    2584#  2584    2586#  2586    2588#  2588    2593#  2593    2595#  2595    2597#  2597
2599#  2599    2601#  2601    2603#  2603    2605#  2605    2607#  2607    2609#  2609    2611#  2611
2613#  2613    2615#  2615    2709#  2709    2711#  2711    2713#  2713    2715#  2715    2717#  2717
2719#  2719    2721#  2721    2723#  2723    2725#  2725    2727#  2727    2729#  2729    2731#  2731
2733#  2733    2735#  2735    2737#  2737    2739#  2739    2741#  2741    2743#  2743    2745#  2745
2747#  2747    2749#  2749    2751#  2751    2753#  2753    2755#  2755    2757#  2757    2759#  2759
2761#  2761    2763#  2763    2765#  2765    2767#  2767    2769#  2769    2771#  2771    2773#  2773
2775#  2775    2777#  2777    2779#  2779    2784#  2784    2786#  2786    2788#  2788    2790#  2790
2792#  2792    2794#  2794    2796#  2796    2798#  2798    2800#  2800    2802#  2802    2804#  2804
2806#  2806    2808#  2808    2810#  2810    2812#  2812    2814#  2814    2816#  2816    2818#  2818
2820#  2820    2822#  2822    2824#  2824    2826#  2826    2828#  2828    2830#  2830    2835#  2835
2837#  2837    2839#  2839    2841#  2841    2843#  2843    2845#  2845    2847#  2847    2849#  2849
2851#  2851    2853#  2853    2855#  2855    2857#  2857    2859#  2859    2861#  2861    2955#  2955
2957#  2957    2959#  2959    2961#  2961    2963#  2963    2965#  2965    2967#  2967    2969#  2969
2971#  2971    2973#  2973    2975#  2975    2977#  2977    2979#  2979    2981#  2981    2983#  2983
2985#  2985    2987#  2987    2989#  2989    2991#  2991    2993#  2993    2995#  2995    2997#  2997
2999#  2999    3001#  3001    3003#  3003    3008#  3008    3010#  3010    3012#  3012    3014#  3014
3016#  3016    3018#  3018    3020#  3020    3022#  3022    3024#  3024    3026#  3026    3028#  3028
3030#  3030    3032#  3032    3034#  3034    3036#  3036    3038#  3038    3040#  3040    3042#  3042
3044#  3044    3046#  3046    3048#  3048    3050#  3050    3052#  3052    3054#  3054    3056#  3056
3058#  3058    3060#  3060    3062#  3062    3064#  3064    3066#  3066    3071#  3071    3073#  3073
3075#  3075    3077#  3077    3079#  3079    3081#  3081    3083#  3083    3085#  3085    3087#  3087
3089#  3089    3091#  3091    3093#  3093    3095#  3095    3097#  3097    3099#  3099    3101#  3101
3103#  3103    3105#  3105    3107#  3107    3109#  3109    3111#  3111    3113#  3113    3115#  3115
3117#  3117    3119#  3119    3121#  3121    3123#  3123    3125#  3125    3127#  3127    3129#  3129
3131#  3131    3133#  3133    3135#  3135    3137#  3137    3139#  3139    3141#  3141    3143#  3143
3145#  3145    3147#  3147    3149#  3149    3243#  3243    3245#  3245    3247#  3247    3250#  3250
3252#  3252    3254#  3254    3256#  3256    3258#  3258    3260#  3260    3262#  3262    3264#  3264
3266#  3266    3268#  3268    3270#  3270    3272#  3272    3274#  3274    3276#  3276    3278#  3278
3280#  3280    3282#  3282    3284#  3284    3286#  3286    3288#  3288    3290#  3290    3292#  3292
3294#  3294    3296#  3296    3298#  3298    3300#  3300    3302#  3302    3304#  3304    3306#  3306
3308#  3308    3310#  3310    3312#  3312    3314#  3314    3316#  3316    3318#  3318    3320#  3320
3325#  3325    3327#  3327    3329#  3329    3331#  3331    3333#  3333    3335#  3335    3337#  3337
3339#  3339    3341#  3341    3343#  3343    3345#  3345    3347#  3347    3349#  3349    3351#  3351
3353#  3353    3355#  3355    3357#  3357    3359#  3359    3361#  3361    3363#  3363    3365#  3365
```

```
          3367#  3367  3369#  3369  3371#  3371  3373#  3373  3375#  3375  3377#  3377  3379#  3379     SEQ 1057
          3384#  3384  3386#  3386  3388#  3388  3390#  3390  3392#  3392  3394#  3394  3396#  3396
          3398#  3398  3400#  3400  3402#  3402  3404#  3404  3406#  3406  3408#  3408  3410#  3410
          3412#  3412  3414#  3414  3416#  3416  3418#  3418  3420#  3420  3422#  3422  3424#  3424
          3426#  3426  3428#  3428  3430#  3430  3432#  3432  3434#  3434  3436#  3436  3438#  3438
          3440#  3440  3442#  3442  3444#  3444  3446#  3446  3448#  3448  3450#  3450  3452#  3452
          3454#  3454  3456#  3456  3458#  3458  3460#  3460  3462#  3462  3464#  3464  3466#  3466
          3468#  3468  3470#  3470  3472#  3472  3474#  3474  3476#  3476  3478#  3478  3480#  3480
          3482#  3482  3484#  3484  3486#  3486  3488#  3488  3490#  3490  3492#  3492  3494#  3494
          3496#  3496  3498#  3498  3500#  3500  3502#  3502  3504#  3504  3506#  3506  3508#  3508
          3510#  3510  3512#  3512  3514#  3514  3516#  3516  3724#  3724  3726#  3726  3728#  3728
          3730#  3730  3732#  3732  3734#  3734  3736#  3736  3738#  3738  3740#  3740  3742#  3742
          3747#  3747  3749#  3749  3751#  3751  3753#  3753  3755#  3755  3757#  3757  3759#  3759
          3761#  3761  3766#  3766  3768#  3768  3770#  3770  3772#  3772  3777#  3777  3779#  3779
          3781#  3781  3783#  3783  3785#  3785  3787#  3787  3789#  3789  3791#  3791  3796#  3796
          3798#  3798  3800#  3800  3802#  3802  3804#  3804  3806#  3806  3808#  3808  3810#  3810
          3812#  3812  3817#  3817  3819#  3819  3821#  3821  3823#  3823  3825#  3825  3830#  3830
          3832#  3832  3834#  3834  3839#  3839  3841#  3841  3843#  3843  4055#  4055  4057#  4057
          4059#  4059  4061#  4061  4063#  4063  4065#  4065  4067#  4067  4069#  4069  4071#  4071
          4073#  4073  4078#  4078  4080#  4080  4082#  4082  4084#  4084  4086#  4086  4088#  4088
          4090#  4090  4092#  4092  4097#  4097  4099#  4099  4101#  4101  4103#  4103  4108#  4108
          4110#  4110  4112#  4112  4114#  4114  4116#  4116  4118#  4118  4120#  4120  4122#  4122
          4127#  4127  4129#  4129  4131#  4131  4133#  4133  4135#  4135  4137#  4137  4139#  4139
          4141#  4141  4143#  4143  4148#  4148  4150#  4150  4152#  4152  4154#  4154  4156#  4156
          4161#  4161  4163#  4163  4165#  4165  4170#  4170  4172#  4172  4174#  4174  4385#  4385
          4387#  4387  4389#  4389  4391#  4391  4393#  4393  4395#  4395  4397#  4397  4402#  4402
          4404#  4404  4406#  4406  4408#  4408  4410#  4410  4412#  4412  4414#  4414  4419#  4419
          4421#  4421  4423#  4423  4425#  4425  4430#  4430  4432#  4432  4434#  4434  4436#  4436
          4438#  4438  4440#  4440  4442#  4442  4447#  4447  4449#  4449  4451#  4451  4453#  4453
          4455#  4455  4457#  4457  4459#  4459  4461#  4461  4466#  4466  4468#  4468  4470#  4470
          4472#  4472  4474#  4474  4479#  4479  4481#  4481  4483#  4483  4488#  4488  4490#  4490
          4492#  4492  4717#  4717  4719#  4719  4721#  4721  4723#  4723  4725#  4725  4730#  4730
          4732#  4732  4734#  4734  4736#  4736  4738#  4738  4740#  4740  4742#  4742  4747#  4747
          4749#  4749  4751#  4751  4753#  4753  4758#  4758  4760#  4760  4765#  4765  4767#  4767
          4769#  4769  4774#  4774  4776#  4776  4778#  4778  4780#  4780  4782#  4782  4787#  4787
          4789#  4789  4791#  4791  4796#  4796  4798#  4798  4800#  4800  5009#  5009  5011#  5011
          5013#  5013  5015#  5015  5017#  5017  5019#  5019  5021#  5021  5026#  5026  5028#  5028
          5030#  5030  5032#  5032  5034#  5034  5036#  5036  5038#  5038  5043#  5043  5045#  5045
          5047#  5047  5049#  5049  5054#  5054  5056#  5056  5058#  5058  5060#  5060  5062#  5062
          5064#  5064  5066#  5066  5071#  5071  5073#  5073  5075#  5075  5077#  5077  5079#  5079
          5081#  5081  5083#  5083  5085#  5085  5090#  5090  5092#  5092  5094#  5094  5096#  5096
          5098#  5098  5103#  5103  5105#  5105  5107#  5107  5112#  5112  5114#  5114  5116#  5116
          5328#  5328  5330#  5330  5332#  5332  5334#  5334  5336#  5336  5341#  5341  5343#  5343
          5345#  5345  5347#  5347  5349#  5349  5351#  5351  5353#  5353  5358#  5358  5360#  5360
          5362#  5362  5364#  5364  5369#  5369  5371#  5371  5376#  5376  5378#  5378  5380#  5380
          5385#  5385  5387#  5387  5389#  5389  5391#  5391  5393#  5393  5398#  5398  5400#  5400
          5402#  5402  5407#  5407  5409#  5409  5411#  5411
%MR        198#   198   200#   200   202#   202   204#   204   209#   209   211#   211   213#   213
           215#   215   220#   220   222#   222   224#   224   226#   226   231#   231   233#   233
           235#   235   237#   237   242#   242   244#   244   246#   246   248#   248   253#   253
           255#   255   257#   257   259#   259   264#   264   266#   266   268#   268   270#   270
           275#   275   277#   277   279#   279   281#   281   283#   283   285#   285   287#   287
           391#   391   393#   393   395#   395   397#   397   402#   402   404#   404   406#   406
```

```
  408#   408    413#   413    415#   415    417#   417    419#   419    424#   424    426#   426
  428#   428    430#   430    435#   435    437#   437    439#   439    441#   441    446#   446
  448#   448    450#   450    452#   452    457#   457    459#   459    461#   461    463#   463
  468#   468    470#   470    472#   472    474#   474    476#   476    478#   478    480#   480
  482#   482    484#   484    486#   486    488#   488    490#   490    492#   492    494#   494
  499#   499    501#   501    503#   503    505#   505    510#   510    512#   512    514#   514
  516#   516    521#   521    523#   523    525#   525    527#   527    532#   532    534#   534
  536#   536    538#   538    543#   543    545#   545    547#   547    549#   549    554#   554
  556#   556    558#   558    560#   560    565#   565    567#   567    569#   569    571#   571
  576#   576    578#   578    580#   580    582#   582    584#   584    586#   586    588#   588
  590#   590    592#   592    594#   594    596#   596    598#   598    600#   600    602#   602
  607#   607    609#   609    611#   611    613#   613    615#   615    617#   617    619#   619
  621#   621    623#   623    625#   625    627#   627    629#   629    631#   631    633#   633
  739#   739    741#   741    743#   743    745#   745    829#   829    831#   831    833#   833
  835#   835    919#   919    921#   921    923#   923    925#   925   1018#  1018   1020#  1020
 1022#  1022   1024#  1024   1117#  1117   1119#  1119   1121#  1121   1123#  1123   1216#  1216
 1218#  1218   1220#  1220   1222#  1222   1315#  1315   1317#  1317   1319#  1319   1321#  1321
 1416#  1416   1418#  1418   1420#  1420   1422#  1422   1577#  1577   1579#  1579   1581#  1581
 1584#  1584   1586#  1586   1588#  1588   1593#  1593   1595#  1595   1597#  1597   1600#  1600
 1602#  1602   1604#  1604   1609#  1609   1611#  1611   1613#  1613   1616#  1616   1618#  1618
 1620#  1620   1625#  1625   1627#  1627   1629#  1629   1632#  1632   1634#  1634   1636#  1636
 1641#  1641   1643#  1643   1645#  1645   1648#  1648   1650#  1650   1652#  1652   1657#  1657
 1659#  1659   1661#  1661   1664#  1664   1666#  1666   1668#  1668   1673#  1673   1675#  1675
 1677#  1677   1680#  1680   1682#  1682   1684#  1684   1689#  1689   1691#  1691   1693#  1693
 1696#  1696   1698#  1698   1700#  1700   1705#  1705   1707#  1707   1709#  1709   1712#  1712
 1714#  1714   1716#  1716   1721#  1721   1723#  1723   1725#  1725   1728#  1728   1730#  1730
 1732#  1732   1737#  1737   1739#  1739   1741#  1741   1744#  1744   1746#  1746   1748#  1748
 1753#  1753   1755#  1755   1757#  1757   1760#  1760   1762#  1762   1764#  1764   1769#  1769
 1771#  1771   1773#  1773   1776#  1776   1778#  1778   1780#  1780   1785#  1785   1787#  1787
 1789#  1789   1792#  1792   1794#  1794   1796#  1796   1801#  1801   1803#  1803   1805#  1805
 1808#  1808   1810#  1810   1812#  1812   1817#  1817   1819#  1819   1821#  1821   1824#  1824
 1826#  1826   1828#  1828   1912#  1912   1914#  1914   1916#  1916   1919#  1919   1921#  1921
 1923#  1923   1925#  1925   2051#  2051   2053#  2053   2055#  2055   2057#  2057   2059#  2059
 2061#  2061   2063#  2063   2065#  2065   2067#  2067   2069#  2069   2071#  2071   2073#  2073
 2075#  2075   2077#  2077   2079#  2079   2081#  2081   2083#  2083   2088#  2088   2090#  2090
 2092#  2092   2094#  2094   2096#  2096   2098#  2098   2100#  2100   2102#  2102   2104#  2104
 2106#  2106   2108#  2108   2110#  2110   2112#  2112   2114#  2114   2116#  2116   2118#  2118
 2120#  2120   2245#  2245   2247#  2247   2249#  2249   2251#  2251   2253#  2253   2255#  2255
 2257#  2257   2259#  2259   2261#  2261   2263#  2263   2265#  2265   2267#  2267   2269#  2269
 2271#  2271   2273#  2273   2275#  2275   2277#  2277   2279#  2279   2281#  2281   2283#  2283
 2285#  2285   2287#  2287   2289#  2289   2291#  2291   2293#  2293   2295#  2295   2297#  2297
 2299#  2299   2301#  2301   2303#  2303   2305#  2305   2307#  2307   2309#  2309   2311#  2311
 2313#  2313   2318#  2318   2320#  2320   2322#  2322   2324#  2324   2326#  2326   2328#  2328
 2330#  2330   2332#  2332   2334#  2334   2336#  2336   2338#  2338   2340#  2340   2342#  2342
 2344#  2344   2346#  2346   2348#  2348   2350#  2350   2352#  2352   2357#  2357   2359#  2359
 2361#  2361   2363#  2363   2365#  2365   2367#  2367   2369#  2369   2371#  2371   2373#  2373
 2375#  2375   2377#  2377   2379#  2379   2381#  2381   2475#  2475   2477#  2477   2479#  2479
 2481#  2481   2483#  2483   2485#  2485   2487#  2487   2489#  2489   2491#  2491   2493#  2493
 2495#  2495   2497#  2497   2499#  2499   2501#  2501   2503#  2503   2505#  2505   2507#  2507
 2509#  2509   2511#  2511   2513#  2513   2515#  2515   2517#  2517   2519#  2519   2521#  2521
 2523#  2523   2525#  2525   2527#  2527   2529#  2529   2531#  2531   2533#  2533   2535#  2535
 2537#  2537   2539#  2539   2541#  2541   2543#  2543   2545#  2545   2550#  2550   2552#  2552
 2554#  2554   2556#  2556   2558#  2558   2560#  2560   2562#  2562   2564#  2564   2566#  2566
```

```
2568#  2568   2570#  2570   2572#  2572   2574#  2574   2576#  2576   2578#  2578   2580#  2580
2582#  2582   2584#  2584   2586#  2586   2588#  2588   2593#  2593   2595#  2595   2597#  2597
2599#  2599   2601#  2601   2603#  2603   2605#  2605   2607#  2607   2609#  2609   2611#  2611
2613#  2613   2615#  2615   2709#  2709   2711#  2711   2713#  2713   2715#  2715   2717#  2717
2719#  2719   2721#  2721   2723#  2723   2725#  2725   2727#  2727   2729#  2729   2731#  2731
2733#  2733   2735#  2735   2737#  2737   2739#  2739   2741#  2741   2743#  2743   2745#  2745
2747#  2747   2749#  2749   2751#  2751   2753#  2753   2755#  2755   2757#  2757   2759#  2759
2761#  2761   2763#  2763   2765#  2765   2767#  2767   2769#  2769   2771#  2771   2773#  2773
2775#  2775   2777#  2777   2779#  2779   2784#  2784   2786#  2786   2788#  2788   2790#  2790
2792#  2792   2794#  2794   2796#  2796   2798#  2798   2800#  2800   2802#  2802   2804#  2804
2806#  2806   2808#  2808   2810#  2810   2812#  2812   2814#  2814   2816#  2816   2818#  2818
2820#  2820   2822#  2822   2824#  2824   2826#  2826   2828#  2828   2830#  2830   2835#  2835
2837#  2837   2839#  2839   2841#  2841   2843#  2843   2845#  2845   2847#  2847   2849#  2849
2851#  2851   2853#  2853   2855#  2855   2857#  2857   2859#  2859   2861#  2861   2955#  2955
2957#  2957   2959#  2959   2961#  2961   2963#  2963   2965#  2965   2967#  2967   2969#  2969
2971#  2971   2973#  2973   2975#  2975   2977#  2977   2979#  2979   2981#  2981   2983#  2983
2985#  2985   2987#  2987   2989#  2989   2991#  2991   2993#  2993   2995#  2995   2997#  2997
2999#  2999   3001#  3001   3003#  3003   3008#  3008   3010#  3010   3012#  3012   3014#  3014
3016#  3016   3018#  3018   3020#  3020   3022#  3022   3024#  3024   3026#  3026   3028#  3028
3030#  3030   3032#  3032   3034#  3034   3036#  3036   3038#  3038   3040#  3040   3042#  3042
3044#  3044   3046#  3046   3048#  3048   3050#  3050   3052#  3052   3054#  3054   3056#  3056
3058#  3058   3060#  3060   3062#  3062   3064#  3064   3066#  3066   3071#  3071   3073#  3073
3075#  3075   3077#  3077   3079#  3079   3081#  3081   3083#  3083   3085#  3085   3087#  3087
3089#  3089   3091#  3091   3093#  3093   3095#  3095   3097#  3097   3099#  3099   3101#  3101
3103#  3103   3105#  3105   3107#  3107   3109#  3109   3111#  3111   3113#  3113   3115#  3115
3117#  3117   3119#  3119   3121#  3121   3123#  3123   3125#  3125   3127#  3127   3129#  3129
3131#  3131   3133#  3133   3135#  3135   3137#  3137   3139#  3139   3141#  3141   3143#  3143
3145#  3145   3147#  3147   3149#  3149   3243#  3243   3245#  3245   3247#  3247   3250#  3250
3252#  3252   3254#  3254   3256#  3256   3258#  3258   3260#  3260   3262#  3262   3264#  3264
3266#  3266   3268#  3268   3270#  3270   3272#  3272   3274#  3274   3276#  3276   3278#  3278
3280#  3280   3282#  3282   3284#  3284   3286#  3286   3288#  3288   3290#  3290   3292#  3292
3294#  3294   3296#  3296   3298#  3298   3300#  3300   3302#  3302   3304#  3304   3306#  3306
3308#  3308   3310#  3310   3312#  3312   3314#  3314   3316#  3316   3318#  3318   3320#  3320
3325#  3325   3327#  3327   3329#  3329   3331#  3331   3333#  3333   3335#  3335   3337#  3337
3339#  3339   3341#  3341   3343#  3343   3345#  3345   3347#  3347   3349#  3349   3351#  3351
3353#  3353   3355#  3355   3357#  3357   3359#  3359   3361#  3361   3363#  3363   3365#  3365
3367#  3367   3369#  3369   3371#  3371   3373#  3373   3375#  3375   3377#  3377   3379#  3379
3384#  3384   3386#  3386   3388#  3388   3390#  3390   3392#  3392   3394#  3394   3396#  3396
3398#  3398   3400#  3400   3402#  3402   3404#  3404   3406#  3406   3408#  3408   3410#  3410
3412#  3412   3414#  3414   3416#  3416   3418#  3418   3420#  3420   3422#  3422   3424#  3424
3426#  3426   3428#  3428   3430#  3430   3432#  3432   3434#  3434   3436#  3436   3438#  3438
3440#  3440   3442#  3442   3444#  3444   3446#  3446   3448#  3448   3450#  3450   3452#  3452
3454#  3454   3456#  3456   3458#  3458   3460#  3460   3462#  3462   3464#  3464   3466#  3466
3468#  3468   3470#  3470   3472#  3472   3474#  3474   3476#  3476   3478#  3478   3480#  3480
3482#  3482   3484#  3484   3486#  3486   3488#  3488   3490#  3490   3492#  3492   3494#  3494
3496#  3496   3498#  3498   3500#  3500   3502#  3502   3504#  3504   3506#  3506   3508#  3508
3510#  3510   3512#  3512   3514#  3514   3516#  3516   3724#  3724   3726#  3726   3728#  3728
3730#  3730   3732#  3732   3734#  3734   3736#  3736   3738#  3738   3740#  3740   3742#  3742
3747#  3747   3749#  3749   3751#  3751   3753#  3753   3755#  3755   3757#  3757   3759#  3759
3761#  3761   3766#  3766   3768#  3768   3770#  3770   3772#  3772   3777#  3777   3779#  3779
3781#  3781   3783#  3783   3785#  3785   3787#  3787   3789#  3789   3791#  3791   3796#  3796
3798#  3798   3800#  3800   3802#  3802   3804#  3804   3806#  3806   3808#  3808   3810#  3810
3812#  3812   3817#  3817   3819#  3819   3821#  3821   3823#  3823   3825#  3825   3830#  3830
```

```
      3832#  3832   3834#  3834   3839#  3839   3841#  3841   3843#  3843   4055#  4055   4057#  4057      SEQ 1060
      4059#  4059   4061#  4061   4063#  4063   4065#  4065   4067#  4067   4069#  4069   4071#  4071
      4073#  4073   4078#  4078   4080#  4080   4082#  4082   4084#  4084   4086#  4086   4088#  4088
      4090#  4090   4092#  4092   4097#  4097   4099#  4099   4101#  4101   4103#  4103   4108#  4108
      4110#  4110   4112#  4112   4114#  4114   4116#  4116   4118#  4118   4120#  4120   4122#  4122
      4127#  4127   4129#  4129   4131#  4131   4133#  4133   4135#  4135   4137#  4137   4139#  4139
      4141#  4141   4143#  4143   4148#  4148   4150#  4150   4152#  4152   4154#  4154   4156#  4156
      4161#  4161   4163#  4163   4165#  4165   4170#  4170   4172#  4172   4174#  4174   4385#  4385
      4387#  4387   4389#  4389   4391#  4391   4393#  4393   4395#  4395   4397#  4397   4402#  4402
      4404#  4404   4406#  4406   4408#  4408   4410#  4410   4412#  4412   4414#  4414   4419#  4419
      4421#  4421   4423#  4423   4425#  4425   4430#  4430   4432#  4432   4434#  4434   4436#  4436
      4438#  4438   4440#  4440   4442#  4442   4447#  4447   4449#  4449   4451#  4451   4453#  4453
      4455#  4455   4457#  4457   4459#  4459   4461#  4461   4466#  4466   4468#  4468   4470#  4470
      4472#  4472   4474#  4474   4479#  4479   4481#  4481   4483#  4483   4488#  4488   4490#  4490
      4492#  4492   4717#  4717   4719#  4719   4721#  4721   4723#  4723   4725#  4725   4730#  4730
      4732#  4732   4734#  4734   4736#  4736   4738#  4738   4740#  4740   4742#  4742   4747#  4747
      4749#  4749   4751#  4751   4753#  4753   4758#  4758   4760#  4760   4765#  4765   4767#  4767
      4769#  4769   4774#  4774   4776#  4776   4778#  4778   4780#  4780   4782#  4782   4787#  4787
      4789#  4789   4791#  4791   4796#  4796   4798#  4798   4800#  4800   5009#  5009   5011#  5011
      5013#  5013   5015#  5015   5017#  5017   5019#  5019   5021#  5021   5026#  5026   5028#  5028
      5030#  5030   5032#  5032   5034#  5034   5036#  5036   5038#  5038   5043#  5043   5045#  5045
      5047#  5047   5049#  5049   5054#  5054   5056#  5056   5058#  5058   5060#  5060   5062#  5062
      5064#  5064   5066#  5066   5071#  5071   5073#  5073   5075#  5075   5077#  5077   5079#  5079
      5081#  5081   5083#  5083   5085#  5085   5090#  5090   5092#  5092   5094#  5094   5096#  5096
      5098#  5098   5103#  5103   5105#  5105   5107#  5107   5112#  5112   5114#  5114   5116#  5116
      5328#  5328   5330#  5330   5332#  5332   5334#  5334   5336#  5336   5341#  5341   5343#  5343
      5345#  5345   5347#  5347   5349#  5349   5351#  5351   5353#  5353   5358#  5358   5360#  5360
      5362#  5362   5364#  5364   5369#  5369   5371#  5371   5376#  5376   5378#  5378   5380#  5380
      5385#  5385   5387#  5387   5389#  5389   5391#  5391   5393#  5393   5398#  5398   5400#  5400
      5402#  5402   5407#  5407   5409#  5409   5411#  5411
.LA   1577   1579   1581   1584   1586   1588   1593   1595   1597   1600   1602   1604   1609   1611
      1613   1616   1618   1620   1625   1627   1629   1632   1634   1636   1641   1643   1645   1648
      1650   1652   1657   1659   1661   1664   1666   1668   1673   1675   1677   1680   1682   1684
      1689   1691   1693   1696   1698   1700   1705   1707   1709   1712   1714   1716   1721   1723
      1725   1728   1730   1732   1737   1739   1741   1744   1746   1748   1753   1755   1757   1760
      1762   1764   1769   1771   1773   1776   1778   1780   1785   1787   1789   1792   1794   1796
      1801   1803   1805   1808   1810   1812   1817   1819   1821   1824   1826   1828   2051   2053
      2055   2057   2059   2061   2063   2065   2067   2069   2071   2073   2075   2077   2079   2081
      2083   2088   2090   2092   2094   2096   2098   2100   2102   2104   2106   2108   2110   2112
      2114   2116   2118   2120   2479   2483   2487   2491   2495   2499   2503   2507   2511   2515
      2519   2523   2527   2531   2535   2539   2543   2552   2554   2558   2562   2566   2570   2574
      2578   2582   2586   2588   2717   2721   2725   2729   2733   2737   2741   2745   2749   2753
      2757   2761   2765   2769   2773   2777   2779   2792   2796   2800   2804   2810   2816   2822
      2828   2830   2857   2859   2965   2969   2973   2977   2983   2989   2995   3001   3003   3083
      3087   3[illegible]89   3091   3093   3095   3097   3101   3103   3107   3109   3111   3113   3115   3117
      3119   3121   3123   3125   3129   3131   3135   3137   3141   3143   3147   3260   3262   3264
      3266   3268   3272   3276   3280   3284   3286   3288   3290   3292   3294   3296   3300   3302
      3306   3308   3312   3314   3318   3333   3337   3343   3349   3353   3357   3363   3369   3373
      3377   3379   3420   3422   3424   3426   3428   3430   3432   3434   3436   3438   3440   3442
      3444   3446   3448   3450   3452   3454   3456   3458   3460   3462   3464   3466   3468   3470
      3472   3474   3476   3478   3480   3482   3484   3486   3488   3490   3492   3494   3496   3498
      3500   3502   3504   3506   3508   3510   3512   3514   3732   3738   3742   3755   3757   3759
      3761   3779   3781   3783   3785   3787   3798   3800   3802   3804   3806   3808   3823   3825
```

```
             3834   3843   4063   4069   4073   4086   4088   4090   4092   4110   4112   4114   4116   4118        SEQ 1061
             4129   4131   4133   4135   4137   4139   4154   4156   4165   4174   4397   4410   4412   4414
             4438   4442   4451   4453   4457   4461   4472   4474   4483   4492   4725   4738   4740   4742
             4760   4767   4769   4780   4782   4791   4800   5021   5034   5036   5038   5062   5066   5075
             5077   5081   5085   5096   5098   5107   5116   5336   5349   5351   5353   5371   5378   5380
             5391   5393   5402   5411
.LADDR        198    391    739    829    919   1018   1117   1216   1315   1416   1577   1912   2051   2245
             2475   2709   2955   3243   3724   3830   4055   4161   4385   4479   4717   4747   4774   4787
             5009   5103   5328   5358   5385   5398
.LAND         198    200    202    204    209    211    213    215    220    222    224    226    231    233
              235    237    242    244    246    248    253    255    257    259    264    266    268    270
              829    833   1577   1593   1609   1625   1641   1657   1673   1689   1705   1721   1737   1753
             1769   1785   1801   1817   1912   1919   2245   2318   2357   2475   2477   2481   2485   2489
             2493   2497   2501   2505   2509   2513   2517   2521   2525   2529   2535   2537   2541   2550
             2556   2560   2564   2568   2572   2576   2580   2584   2593   2599   2601   2603   2605   2607
             2609   2611   2613   2615   2709   2711   2713   2784   2835   2861   2955   2957   3008   3010
             3071   3073   3083   3087   3091   3095   3101   3107   3111   3115   3119   3123   3129   3135
             3141   3147   3243   3325   3327   3333   3337   3339   3343   3345   3349   3353   3357   3359
             3363   3365   3369   3373   3377   3384   3484   3486   3488   3490   3492   3494   3496   3498
             3500   3502   3504   3506   3508   3510   3512   3514   3724   3726   3734   3747   3759   3766
             3819   4055   4057   4065   4078   4090   4097   4150   4385   4393   4402   4419   4468   4717
             4721   4730   4747   4776   5009   5017   5026   5043   5092   5328   5332   5341   5358   5387
.LB          1577   1579   1584   1586   1593   1595   1600   1602   1609   1611   1616   1618   1625   1627
             1632   1634   1641   1643   1648   1650   1657   1659   1664   1666   1673   1675   1680   1682
             1689   1691   1696   1698   1705   1707   1712   1714   1721   1723   1728   1730   1737   1739
             1744   1746   1753   1755   1760   1762   1769   1771   1776   1778   1785   1787   1792   1794
             1801   1803   1808   1810   1817   1819   1824   1826   2051   2053   2055   2057   2059   2061
             2063   2065   2067   2069   2071   2073   2075   2077   2079   2081   2083   2475   2481   2485
             2489   2493   2497   2501   2505   2509   2513   2517   2521   2525   2550   2556   2558   2560
             2562   2564   2566   2568   2570   2572   2574   2576   2578   2580   2582   2584   2586   2711
             2715   2717   2719   2721   2723   2725   2727   2729   2731   2733   2735   2737   2739   2741
             2743   2745   2747   2749   2751   2753   2755   2757   2759   2761   2763   2765   2767   2769
             2771   2773   2775   2777   2790   2792   2794   2796   2798   2800   2802   2804   2806   2808
             2810   2812   2814   2816   2818   2820   2822   2824   2826   2828   2839   2847   2849   2853
             2855   2857   2859   2957   2963   2965   2967   2969   2971   2973   2975   2977   2979   2981
             2983   2985   2987   2989   2991   2993   2995   2997   2999   3001   3038   3040   3042   3044
             3073   3077   3079   3081   3083   3085   3087   3089   3091   3093   3095   3097   3099   3101
             3103   3105   3107   3109   3111   3113   3115   3117   3119   3121   3123   3125   3127   3129
             3131   3133   3135   3137   3139   3141   3143   3145   3147   3245   3270   3274   3278   3282
             3296   3298   3302   3304   3308   3310   3314   3316   3329   3331   3333   3335   3337   3339
             3341   3343   3345   3347   3349   3351   3353   3355   3357   3359   3361   3363   3365   3367
             3369   3371   3373   3375   3377   3388   3390   3392   3394   3396   3398   3400   3402   3404
             3406   3408   3410   3412   3414   3416   3420   3422   3426   3428   3432   3434   3438   3440
             3444   3446   3452   3454   3456   3458   3460   3462   3464   3466   3468   3470   3472   3474
             3476   3478   3480   3482   3726   3728   3730   3732   3736   3738   3740   3749   3751   3755
             3757   3759   3777   3779   3781   3783   3785   3787   3796   3798   3800   3802   3804   3806
             3808   3817   3823   3830   3832   3839   3841   4057   4059   4061   4063   4067   4069   4071
             4080   4082   4086   4088   4090   4108   4110   4112   4114   4116   4118   4127   4129   4131
             4133   4135   4137   4139   4148   4154   4161   4163   4170   4172   4387   4389   4391   4395
             4397   4404   4406   4410   4412   4414   4430   4432   4436   4438   4440   4442   4447   4451
             4453   4455   4457   4459   4461   4468   4470   4472   4479   4481   4490   4719   4723   4725
             4732   4734   4738   4740   4742   4758   4765   4767   4776   4778   4780   4787   4789   4798
             5011   5013   5015   5019   5021   5028   5030   5034   5036   5038   5054   5056   5060   5062
```

```
        5064  5066  5071  5075  5077  5079  5081  5083  5085  5092  5094  5096  5103  5105    SEQ 1062
        5114  5330  5334  5336  5343  5345  5349  5351  5353  5369  5376  5378  5387  5389
        5391  5398  5400  5409
.LCONT  1579  1586  1595  1602  1611  1618  1627  1634  1643  1650  1659  1666  1675  1682
        1691  1698  1707  1714  1723  1730  1739  1746  1755  1762  1771  1778  1787  1794
        1803  1810  1819  1826  1914  1921  1923  2053  2055  2057  2059  2061  2063  2065
        2067  2069  2071  2073  2075  2077  2079  2081  2083  2247  2249  2251  2253  2255
        2257  2259  2261  2263  2265  2267  2269  2271  2273  2275  2277  2279  2281  2283
        2285  2287  2289  2291  2293  2295  2297  2299  2301  2303  2305  2307  2309  2320
        2322  2324  2326  2328  2330  2332  2334  2336  2338  2340  2342  2344  2346  2348
        2359  2361  2363  2365  2367  2369  2371  2373  2375  2377  2477  2479  2481  2483
        2485  2487  2489  2491  2493  2495  2497  2499  2501  2503  2505  2507  2509  2511
        2513  2515  2517  2519  2521  2523  2525  2527  2529  2531  2533  2535  2537  2539
        2541  2543  2552  2554  2556  2558  2560  2562  2564  2566  2568  2570  2572  2574
        2576  2578  2580  2582  2584  2586  2595  2597  2599  2601  2603  2605  2607  2609
        2611  2613  2711  2713  2715  2717  2719  2721  2723  2725  2727  2729  2731  2733
        2735  2737  2739  2741  2743  2745  2747  2749  2751  2753  2755  2757  2759  2761
        2763  2765  2767  2769  2771  2773  2775  2777  2786  2788  2790  2792  2794  2796
        2798  2800  2802  2804  2806  2808  2810  2812  2814  2816  2818  2820  2822  2824
        2826  2828  2837  2839  2841  2843  2845  2847  2849  2851  2853  2855  2857  2859
        2957  2959  2961  2963  2965  2967  2969  2971  2973  2975  2977  2979  2981  2983
        2985  2987  2989  2991  2993  2995  2997  2999  3001  3010  3012  3014  3016  3018
        3020  3022  3024  3026  3028  3030  3032  3034  3036  3038  3040  3042  3044  3046
        3048  3050  3052  3054  3056  3058  3060  3062  3064  3073  3075  3077  3079  3081
        3083  3085  3087  3089  3091  3093  3095  3097  3099  3101  3103  3105  3107  3109
        3111  3113  3115  3117  3119  3121  3123  3125  3127  3129  3131  3133  3135  3137
        3139  3141  3143  3145  3147  3245  3247  3250  3252  3254  3256  3258  3260  3262
        3264  3266  3268  3270  3272  3274  3276  3278  3280  3282  3284  3286  3288  3290
        3292  3294  3296  3298  3300  3302  3304  3306  3308  3310  3312  3314  3316  3318
        3327  3329  3331  3333  3335  3337  3339  3341  3343  3345  3347  3349  3351  3353
        3355  3357  3359  3361  3363  3365  3367  3369  3371  3373  3375  3377  3386  3388
        3390  3392  3394  3396  3398  3400  3402  3404  3406  3408  3410  3412  3414  3416
        3418  3420  3422  3424  3426  3428  3430  3432  3434  3436  3438  3440  3442  3444
        3446  3448  3450  3452  3454  3456  3458  3460  3462  3464  3466  3468  3470  3472
        3474  3476  3478  3480  3482  3484  3486  3488  3490  3492  3494  3496  3498  3500
        3502  3504  3506  3508  3510  3512  3514  3726  3728  3730  3732  3734  3736  3738
        3740  3749  3751  3753  3755  3757  3759  3772  3777  3779  3781  3783  3785  3787
        3789  3798  3800  3802  3804  3806  3808  3810  3819  3821  3823  3832  3841  4057
        4059  4061  4063  4065  4067  4069  4071  4080  4082  4084  4086  4088  4090  4103
        4108  4110  4112  4114  4116  4118  4120  4129  4131  4133  4135  4137  4139  4141
        4150  4152  4154  4163  4172  4387  4389  4391  4393  4395  4404  4406  4408  4410
        4412  4425  4430  4432  4434  4436  4438  4440  4449  4451  4453  4455  4457  4459
        4468  4470  4472  4481  4490  4719  4721  4723  4732  4734  4736  4738  4740  4753
        4758  4767  4776  4778  4780  4789  4798  5011  5013  5015  5017  5019  5028  5030
        5032  5034  5036  5049  5054  5056  5058  5060  5062  5064  5073  5075  5077  5079
        5081  5083  5092  5094  5096  5105  5114  5330  5332  5334  5343  5345  5347  5349
        5351  5364  5369  5378  5387  5389  5391  5400  5409
.LCRY   1315  1319  2263  2265  2267  2269  2271  2273  2275  2277  2291  2303  2330  2334
        2336  2340  2342  2363  2367  2373  2489  2493  2497  2501  2517  2521  2533  2541
        2556  2560  2564  2568  2580  2584  2601  2603  2605  2607  2715  2719  2723  2727
        2739  2747  2759  2767  2802  2808  2826  2843  2845  2847  2849  2967  2971  2975
        2981  3014  3016  3018  3022  3032  3042  3050  3052  3062  3081  3085  3089  3093
        3105  3117  3121  3133  3139  3250  3252  3254  3256  3258  3288  3290  3316  3335
        3337  3339  3341  3343  3345  3347  3768  3821  4059  4061  4067  4071  4086  4088      SEQ 1063
        4099  4108  4110  4112  4114  4116  4118  4120  4129  4131  4133  4135  4137  4139
        4141  4152  4154  4163  4172  4421  4470  4719  4723  4738  4740  4749  4758  4767
        4778  4780  4789  4798  5045  5094  5330  5334  5349  5351  5360  5369  5378  5389
        5391  5400  5409
.LD      198   200   202   204   209   211   213   215   220   222   224   226   231   233
         235   237   242   244   246   248   253   255   257   259   264   266   268   270
         275   277   279   281   283   285   287   391   393   395   397   402   404   406
         408   413   415   417   419   424   426   428   430   435   437   439   441   446
         448   450   452   457   459   461   463   468   470   472   474   476   478   480
         482   484   486   488   490   492   494   499   501   503   505   510   512   514
         516   521   523   525   527   532   534   536   538   543   545   547   549   554
         556   558   560   565   567   569   571   576   578   580   582   584   586   588
         590   592   594   596   598   600   602   607   609   611   613   615   617   619
         621   623   625   627   629   631   633   739   741   743   745   829   831   833
         835   919   921   923   925  1018  1020  1022  1024  1117  1119  1121  1123  1216
        1218  1220  1222  1315  1317  1319  1321  1416  1418  1420  1422  1577  1579  1581
        1584  1586  1588  1593  1595  1597  1600  1602  1604  1609  1611  1613  1616  1618
        1620  1625  1627  1629  1632  1634  1636  1641  1643  1645  1648  1650  1652  1657
        1659  1661  1664  1666  1668  1673  1675  1677  1680  1682  1684  1689  1691  1693
        1696  1698  1700  1705  1707  1709  1712  1714  1716  1721  1723  1725  1728  1730
        1732  1737  1739  1741  1744  1746  1748  1753  1755  1757  1760  1762  1764  1769
        1771  1773  1776  1778  1780  1785  1787  1789  1792  1794  1796  1801  1803  1805
        1808  1810  1812  1817  1819  1821  1824  1826  1828  1912  1914  1916  1919  1921
        1923  1925  2051  2053  2055  2057  2059  2061  2063  2065  2067  2069  2071  2073
        2075  2077  2079  2081  2083  2088  2090  2092  2094  2096  2098  2100  2102  2104
        2106  2108  2110  2112  2114  2116  2118  2120  2245  2247  2249  2251  2253  2255
        2257  2259  2261  2263  2265  2267  2269  2271  2273  2275  2277  2279  2281  2283
        2285  2287  2289  [illegible]  2293  2295  2297  2299  2301  2303  2305  2307  2309  2311
        2313  2318  2320  2322  2324  2326  2328  2330  2332  2334  2336  2338  2340  2342
        2344  2346  2348  2350  2352  2357  2359  2361  2363  2365  2367  2369  2371  2373
        2375  2377  2379  2381  2475  2477  2479  2481  2483  2485  2487  2489  2491  2493
        2495  2497  2499  2501  2503  2505  2507  2509  2511  2513  2515  2517  2519  2521
        2523  2525  2527  2529  2531  2533  2535  2537  2539  2541  2543  2545  2550  2552
        2554  2556  2558  2560  2562  2564  2566  2568  2570  2572  2574  2576  2578  2580
        2582  2584  2586  2588  2593  2595  2597  2599  2601  2603  2605  2607  2609  2611
        2613  2615  2709  2711  2713  2715  2717  2719  2721  2723  2725  2727  2729  2731
        2733  2735  2737  2739  2741  2743  2745  2747  2749  2751  2753  2755  2757  2759
        2761  2763  2765  2767  2769  2771  2773  2775  2777  2779  2784  2786  2788  2790
        2792  2794  2796  2798  2800  2802  2804  2806  2808  2810  2812  2814  2816  2818
        2820  2822  2824  2826  2828  2830  2835  2837  2839  2841  2843  2845  2847  2849
        2851  2853  2855  2857  2859  2861  2955  2957  2959  2961  2963  2965  2967  2969
        2971  2973  2975  2977  2979  2981  2983  2985  2987  2989  2991  2993  2995  2997
        2999  3001  3003  3008  3010  3012  3014  3016  3018  3020  3022  3024  3026  3028
        3030  3032  3034  3036  3038  3040  3042  3044  3046  3048  3050  3052  3054  3056
        3058  3060  3062  3064  3066  3071  3073  3075  3077  3079  3081  3083  3085  3087
        3089  3091  3093  3095  3097  3099  3101  3103  3105  3107  3109  3111  3113  3115
        3117  3119  3121  3123  3125  3127  3129  3131  3133  3135  3137  3139  3141  3143
        3145  3147  3149  3243  3245  3247  3250  3252  3254  3256  3258  3260  3262  3264
        3266  3268  3270  3272  3274  3276  3278  3280  3282  3284  3286  3288  3290  3292
        3294  3296  3298  3300  3302  3304  3306  3308  3310  3312  3314  3316  3318  3320
        3325  3327  3329  3331  3333  3335  3337  3339  3341  3343  3345  3347  3349  3351
        3353  3355  3357  3359  3361  3363  3365  3367  3369  3371  3373  3375  3377  3379
```

```
             3384   3386   3388   3390   3392   3394   3396   3398   3400   3402   3404   3406   3408   3410      SEQ 1064
             3412   3414   3416   3418   3420   3422   3424   3426   3428   3430   3432   3434   3436   3438
             3440   3442   3444   3446   3448   3450   3452   3454   3456   3458   3460   3462   3464   3466
             3468   3470   3472   3474   3476   3478   3480   3482   3484   3486   3488   3490   3492   3494
             3496   3498   3500   3502   3504   3506   3508   3510   3512   3514   3516   3724   3726   3728
             3730   3732   3734   3736   3738   3740   3742   3747   3749   3751   3753   3755   3757   3759
             3761   3766   3768   3770   3772   3777   3779   3781   3783   3785   3787   3789   3791   3796
             3798   3800   3802   3804   3806   3808   3810   3812   3817   3819   3821   3823   3825   3830
             3832   3834   3839   3841   3843   4055   4057   4059   4061   4063   4065   4067   4069   4071
             4073   4078   4080   4082   4084   4086   4088   4090   4092   4097   4099   4101   4103   4108
             4110   4112   4114   4116   4118   4120   4122   4127   4129   4131   4133   4135   4137   4139
             4141   4143   4148   4150   4152   4154   4156   4161   4163   4165   4170   4172   4174   4385
             4387   4389   4391   4393   4395   4397   4402   4404   4406   4408   4410   4412   4414   4419
             4421   4423   4425   4430   4432   4434   4436   4438   4440   4442   4447   4449   4451   4453
             4455   4457   4459   4461   4466   4468   4470   4472   4474   4479   4481   4483   4488   4490
             4492   4717   4719   4721   4723   4725   4730   4732   4734   4736   4738   4740   4742   4747
             4749   4751   4753   4758   4760   4765   4767   4769   4774   4776   4778   4780   4782   4787
             4789   4791   4796   4798   4800   5009   5011   5013   5015   5017   5019   5021   5026   5028
             5030   5032   5034   5036   5038   5043   5045   5047   5049   5054   5056   5058   5060   5062
             5064   5066   5071   5073   5075   5077   5079   5081   5083   5085   5090   5092   5094   5096
             5098   5103   5105   5107   5112   5114   5116   5328   5330   5332   5334   5336   5341   5343
             5345   5347   5349   5351   5353   5358   5360   5362   5364   5369   5371   5376   5378   5380
             5385   5387   5389   5391   5393   5398   5400   5402   5407   5409   5411
.LJ           198    200    202    204    209    211    213    215    220    222    224    226    231    233
              235    237    242    244    246    248    253    255    257    259    264    266    268    270
              275    277    279    281    283    285    287    391    393    395    397    402    404    406
              408    413    415    417    419    424    426    428    430    435    437    439    441    446
              448    450    452    457    459    461    463    468    470    472    474    476    478    480
              482    484    486    488    490    492    494    499    501    503    505    510    512    514
              516    521    523    525    527    532    534    536    538    543    545    547    549    554
              556    558    560    565    567    569    571    576    578    580    582    584    586    588
              590    592    594    596    598    600    602    607    609    611    613    615    617    619
              621    623    625    627    629    631    633    739    741    743    745    829    831    833
              835    919    921    923    925   1018   1020   1022   1024   1117   1119   1121   1123   1216
             1218   1220   1222   1315   1317   1319   1321   1416   1418   1420   1422   1577   1581   1584
             1588   1593   1597   1600   1604   1609   1613   1616   1620   1625   1629   1632   1636   1641
             1645   1648   1652   1657   1661   1664   1668   1673   1677   1680   1684   1689   1693   1696
             1700   1705   1709   1712   1716   1721   1725   1728   1732   1737   1741   1744   1748   1753
             1757   1760   1764   1769   1773   1776   1780   1785   1789   1792   1796   1801   1805   1808
             1812   1817   1821   1824   1828   1912   1916   1919   1925   2051   2088   2090   2092   2094
             2096   2098   2100   2102   2104   2106   2108   2110   2112   2114   2116   2118   2120   2245
             2311   2313   2318   2350   2352   2357   2379   2381   2475   2545   2550   2588   2593   2615
             2709   2779   2784   2830   2835   2861   2955   3003   3008   3066   3071   3149   3243   3320
             3325   3379   3384   3516   3724   3742   3747   3761   3766   3768   3770   3791   3796   3812
             3817   3825   3830   3834   3839   3843   4055   4073   4078   4092   4097   4099   4101   4122
             4127   4143   4148   4156   4161   4165   4170   4174   4385   4397   4402   4414   4419   4421
             4423   4442   4447   4461   4466   4474   4479   4483   4488   4492   4717   4725   4730   4742
             4747   4749   4751   4760   4765   4769   4774   4782   4787   4791   4796   4800   5009   5021
             5026   5038   5043   5045   5047   5066   5071   5085   5090   5098   5103   5107   5112   5116
             5328   5336   5341   5353   5358   5360   5362   5371   5376   5380   5385   5393   5398   5402
             5407   5411
.LJMAP        198    200    202    204    209    211    213    215    220    222    224    226    231    233
              235    237    242    244    246    248    253    255    257    259    264    266    268    270
```

```
         275   277   279   281   283   285   287   391   393   395   397   402   404   406
         408   413   415   417   419   424   426   428   430   435   437   439   441   446
         448   450   452   457   459   461   463   468   470   472   474   476   478   480
         482   484   486   488   490   492   494   499   501   503   505   510   512   514
         516   521   523   525   527   532   534   536   538   543   545   547   549   554
         556   558   560   565   567   569   571   576   578   580   582   584   586   588
         590   592   594   596   598   600   602   607   609   611   613   615   617   619
         621   623   625   627   629   631   633   739   741   743   745   829   831   833
         835   919   921   923   925  1018  1020  1022  1024  1117  1119  1121  1123  1216
        1218  1220  1222  1315  1317  1319  1321  1416  1418  1420  1422  1577  1581  1584
        1588  1593  1597  1600  1604  1609  1613  1616  1620  1625  1629  1632  1636  1641
        1645  1648  1652  1657  1661  1664  1668  1673  1677  1680  1684  1689  1693  1696
        1700  1705  1709  1712  1716  1721  1725  1728  1732  1737  1741  1744  1748  1753
        1757  1760  1764  1769  1773  1776  1780  1785  1789  1792  1796  1801  1805  1808
        1812  1817  1821  1824  1828  1912  1916  1919  1925  2051  2088  2090  2092  2094
        2096  2098  2100  2102  2104  2106  2108  2110  2112  2114  2116  2118  2120  2245
        2311  2313  2318  2350  2352  2357  2379  2381  2475  2545  2550  2588  2593  2615
        2709  2779  2784  2830  2835  2861  2955  3003  3008  3066  3071  3149  3243  3320
        3325  3379  3384  3516  3724  3742  3747  3761  3766  3768  3770  3791  3796  3812
        3817  3825  3830  3834  3839  3843  4055  4073  4078  4092  4097  4099  4101  4122
        4127  4143  4148  4156  4161  4165  4170  4174  4385  4397  4402  4414  4419  4421
        4423  4442  4447  4461  4466  4474  4479  4483  4488  4492  4717  4725  4730  4742
        4747  4749  4751  4760  4765  4769  4774  4782  4787  4791  4796  4800  5009  5021
        5026  5038  5043  5045  5047  5066  5071  5085  5090  5098  5103  5107  5112  5116
        5328  5336  5341  5353  5358  5360  5362  5371  5376  5380  5385  5393  5398  5402
        5407  5411
.LMGC    198   200   202   204   209   211   213   215   220   222   224   226   231   233
         235   237   242   244   246   248   253   255   257   259   264   266   268   270
         275   277   279   281   283   285   287   393   397   404   408   415   419   426
         430   437   441   448   452   459   463   470   474   478   482   486   490   494
         501   505   512   516   523   527   534   538   545   549   556   560   567   571
         578   582   586   590   594   598   602   609   613   617   621   625   629   633
         739   741   743   745   829   831   833   835   919   921   923   925  1018  1020
        1022  1024  1117  1119  1121  1123  1216  1218  1220  1222  1315  1317  1319  1321
        1416  1418  1420  1422  1579  1581  1586  1588  1595  1597  1602  1604  1611  1613
        1618  1620  1627  1629  1634  1636  1643  1645  1650  1652  1659  1661  1666  1668
        1675  1677  1682  1684  1691  1693  1698  1700  1707  1709  1714  1716  1723  1725
        1730  1732  1739  1741  1746  1748  1755  1757  1762  1764  1771  1773  1778  1780
        1787  1789  1794  1796  1803  1805  1810  1812  1819  1821  1826  1828  1914  1916
        1923  1925  2051  2053  2055  2057  2059  2061  2063  2065  2067  2069  2071  2073
        2075  2077  2079  2081  2083  2088  2090  2092  2094  2096  2098  2100  2102  2104
        2106  2108  2110  2112  2114  2116  2118  2120  2311  2313  2350  2352  2379  2381
        2545  2588  2615  2779  2830  2861  3003  3066  3149  3320  3379  3516  3742  3761
        3791  3812  3825  3834  3843  4073  4092  4122  4143  4156  4165  4174  4397  4414
        4442  4461  4474  4483  4492  4725  4742  4760  4769  4782  4791  4800  5021  5038
        5066  5085  5098  5107  5116  5336  5353  5371  5380  5393  5402  5411
.LNAND   275   277   279   281   283   285   287   919   923  2715  2719  2723  2727  2731
        2735  2739  2743  2747  2751  2755  2759  2763  2767  2771  2775  2790  2794  2798
        2802  2808  2814  2820  2826  2841  2843  2845  2847  2849  2851  2853  2855  2857
        2859
.LOENA   198   200   202   204   209   211   213   215   220   222   224   226   231   233
         235   237   242   244   246   248   253   255   257   259   264   266   268   270
         275   277   279   281   283   285   287   393   397   404   408   415   419   426
```

```
            430    437    441    448    452    459    463    470    474    478    482    486    490    494
            501    505    512    516    523    527    534    538    545    549    556    560    567    571
            578    582    586    590    594    598    602    609    613    617    621    625    629    633
            741    745    831    835    921    925   1020   1024   1119   1123   1218   1222   1317   1321
           1418   1422   1581   1588   1597   1604   1613   1620   1629   1636   1645   1652   1661   1668
           1677   1684   1693   1700   1709   1716   1725   1732   1741   1748   1757   1764   1773   1780
           1789   1796   1805   1812   1821   1828   1916   1925   2088   2090   2092   2094   2096   2098
           2100   2102   2104   2106   2108   2110   2112   2114   2116   2118   2120   2311   2313   2350
           2352   2379   2381   25[illegible]   2588   2615   2779   2830   2861   3003   3066   3149   3320   3379
           3516   3742   3761   3791   3812   3825   3834   3843   4073   4092   4122   4143   4156   4165
           4174   4397   4414   4442   4461   4474   4483   4492   4725   4742   4760   4769   4782   4791
           4800   5021   5038   5066   5085   5098   5107   5116   5336   5353   5371   5380   5393   5402
           5411
.LOR        393    397    404    408    415    419    426    430    437    441    448    452    459    463
            470    474    478    482    486    490    494    501    505    512    516    523    527    534
            538    545    549    556    560    567    571    578    582    586    590    594    598    602
            609    613    617    621    625    629    633    739    741    743    745    831    835    921
            925   1020   1024   1119   1123   1218   1222   1317   1321   1418   1422   1579   1581   1586
           1588   1595   1597   1602   1604   1611   1613   1618   1620   1627   1629   1634   1636   1643
           1645   1650   1652   1659   1661   1666   1668   1675   1677   1682   1684   1691   1693   1698
           1700   1707   1709   1714   1716   1723   1725   1730   1732   1739   1741   1746   1748   1755
           1757   1762   1764   1771   1773   1778   1780   1787   1789   1794   1796   1803   1805   1810
           1812   1819   1821   1826   1828   1914   1916   1923   1925   2051   2053   2055   2057   2059
           2061   2063   2065   2067   2069   2071   2073   2075   2077   2079   2081   2083   2088   2090
           2092   2094   2096   2098   2100   2102   21[illegible]   2106   2108   2110   2112   2114   2116   2118
           2120   2247   2249   2251   2253   2255   2257   2259   2261   2263   2265   2267   2269   2271
           2273   2275   2277   2279   2281   2283   2285   2287   2289   2291   2293   2295   2297   2299
           2301   2303   2305   2307   2309   2311   2313   2322   2324   2326   2328   2330   2332   2334
           2336   2338   2340   2342   2344   2346   2348   2350   2352   2361   2363   2365   2367   2369
           2371   2373   2375   2377   2379   2381   2479   [illegible]   2487   2491   2495   2499   2503   2507
           2511   2515   2519   2523   2527   2531   2535   [illegible]   2543   2545   2558   2562   2566   2570
           2574   2578   2582   2586   2588   2717   2721   27[illegible]   2729   2733   2737   2741   2745   2749
           2753   2757   2761   2765   2769   2773   2777   2779   2792   2796   2800   2804   2806   2810
           2812   2816   2818   2822   2824   2828   2830   2965   2969   2973   2977   2979   2983   2985
           2989   2991   2995   2997   3001   3003   3066   3097   3103   3125   3131   3137   3143   3149
           3272   3276   3280   3284   3296   3300   3302   3306   3308   3312   3314   3318   3320   3379
           3388   3390   3392   3394   3396   3398   3400   3402   3404   3406   3408   3410   3412   3414
           3416   3418   3516   3732   3738   3742   3761   3770   3772   3791   3812   3825   3830   3834
           3843   4073   4092   4101   4103   4122   4143   4156   4161   4165   4174   4397   4414   4423
           4425   4432   4434   4438   4442   4453   4457   4461   4474   4479   4483   4492   4751   4753
           4760   4769   4782   4787   4791   4800   5021   5038   5047   5049   5056   5058   5062   5066
           5077   5081   5085   5098   5103   5107   5116   5362   5364   5371   5380   5393   5398   5402
           5411
.LPLUS     1216   1220   3728   3730   3736   3740   3755   3757   3768   3777   3779   3781   3783   3785
           3787   3789   3798   3800   3802   3804   3806   3808   3810   3821   3823   3832   3841   4059
           4061   4063   4067   4069   4071   4086   4088   4099   4108   4110   4112   4114   4116   4118
           4120   4129   4131   4133   4135   4137   4139   4141   4152   4154   4163   4172   4421   4470
           4725   4742   4749   4778   5045   5094   5336   5353   5360   5389
.LRMIN      499    503    510    514    521    525    532    536    543    547    554    558    565    569
            576    580    584    588    592    596    600    607    611    615    619    623    627    631
           1315   1319   4387   4389   4391   4395   4410   4412   4430   4436   4440   4451   4455   4459
           4472   4481   4490   4719   4723   4738   4740   4758   4767   4780   4789   4798
.LSOA       204    215    226    237    248    259    270    468    472    476    480    484    488    492
```

```
             576    580    584    588    592    596    600   1581   1584   1588   1597   1600   1604   1613      SEQ 1067
            1616   1620   1629   1632   1636   1645   1648   1652   1661   1664   1668   1677   1680   1684
            1693   1696   1700   1709   1712   1716   1725   1728   1732   1741   1744   1748   1757   1760
            1764   1773   1776   1780   1789   1792   1796   1805   1808   1812   1821   1824   1828   1919
            1921   2088   2090   2092   2094   2096   2098   2100   2102   2104   2106   2108   2110   2112
            2114   2116   2118   2247   2249   2251   2253   2255   2257   2259   2261   2311   2320   2334
            2336   2338   2340   2350   2359   2379   2477   2481   2485   2489   2493   2552   2554   2572
            2576   2580   2584   2588   2595   2597   2763   2767   2771   2775   2779   2786   2788   2806
            2812   2818   2824   2830   2837   2839   2857   2859   2959   2961   2979   2985   2991   2997
            3003   3046   3048   3050   3052   3054   3075   3077   3079   3089   3093   3097   3103   3125
            3131   3137   3143   3245   3247   3286   3288   3290   3292   3294   3296   3302   3308   3314
            3320   3329   3371   3375   3379   3388   3390   3392   3394   3396   3398   3400   3402   3404
            3406   3408   3410   3412   3414   3416   3418   3516   3724   3726   3730   3734   3742   3747
            3749   3751   3753   3761   3766   3768   3770   3772   3777   3779   3781   3796   3798   3800
            3802   3817   3819   3825   3830   3834   3839   3843   4055   4057   4061   4065   4073   4078
            4080   4082   4084   4092   4097   4099   4101   4103   4108   4110   4112   4127   4129   4131
            4133   4148   4150   4156   4161   4165   4170   4174   4385   4387   4393   4402   4404   4406
            4408   4419   4421   4423   4425   4430   4432   4434   4447   4449   4451   4453   4466   4474
            4479   4483   4488   4492   4717   4719   4721   4730   4732   4734   4736   4747   4749   4751
            4753   4758   4760   4765   4767   4769   4774   4782   4787   4791   4796   4800   5009   5011
            5017   5026   5028   5030   5032   5043   5045   5047   5049   5054   5056   5058   5071   5073
            5075   5077   5090   5098   5103   5107   5112   5116   5328   5330   5332   5341   5343   5345
            5347   5358   5360   5362   5364   5369   5371   5376   5378   5380   5385   5393   5398   5402
            5407   5411
.LSOB        202    213    224    235    246    257    268    607    611    615    619    623    627    631
            2295   2297   2299   2301   2303   2305   2307   2309   2342   2344   2346   2348   2529   2533
            2537   2541   2853   2855   3056   3058   3060   3062   3064   3099   3105   3298   3304   3310
            3316   3339   3345   3359   3361   3365   3367   3740   3783   3785   3787   3804   3806   3808
            4071   4114   4116   4118   4135   4137   4139   4389   4436   4455   4468   4470   4776   4778
            5013   5060   5079   5092   5094   5387   5389
.LSOQ        200    211    222    233    244    255    266    391    395    402    406    413    417    424
             428    435    439    446    450    457    461    499    503    510    514    521    525    532
             536    543    547    554    558    565    569    741    745    831    835    921    925   1020
            1024   1119   1123   1218   1222   1317   1321   1418   1422   1914   1916   1923   1925   2120
            2279   2281   2283   2285   2287   2289   2291   2293   2313   2322   2324   2326   2328   2330
            2332   2352   2361   2381   2513   2517   2521   2525   2545   2556   2560   2564   2568   2747
            2751   2755   2759   2851   3038   3040   3042   3044   3081   3085   3149   3270   3274   3278
            3282   3351   3355   3736   3789   3791   3810   3812   3821   4067   4120   4122   4141   4143
            4152   4395   4440   4459   4723   5019   5064   5083   5334
.LSAB       1579   1586   1595   1602   1611   1618   1627   1634   1643   1650   1659   1666   1675   1682
            1691   1698   1707   1714   1723   1730   1739   1746   1755   1762   1771   1778   1787   1794
            1803   1810   1819   1826   2371   2373   2375   2377   2558   2562   2566   2570   2574   2578
            2582   2586   2607   2609   2611   2613   2717   2721   2725   2729   2731   2733   2735   2737
            2739   2741   2743   2745   2749   2753   2757   2761   2765   2769   2773   2777   2792   2796
            2800   2804   2808   2810   2814   2816   2820   2822   2826   2828   2847   2849   2965   2969
            2973   2977   2981   2983   2987   2989   2993   2995   2999   3001   3028   3030   3032   3034
            3036   3083   3087   3091   3095   3101   3107   3111   3115   3119   3123   3127   3129   3133
            3135   3139   3141   3145   3147   3260   3262   3264   3266   3268   3272   3276   3280   3284
            3300   3306   3312   3318   3333   3337   3341   3343   3347   3349   3353   3357   3363   3369
            3373   3377   3386   3452   3454   3456   3458   3460   3462   3464   3466   3468   3470   3472
            3474   3476   3478   3480   3482   3728   3732   3738   3755   3759   3832   3841   4059   4063
            4069   4086   4090   4163   4172   4391   4397   4410   4414   4438   4442   4457   4461   4481
            4725   4738   4742   4789   5015   5021   5034   5038   5062   5066   5081   5085   5105   5336
```

```
          5349    5353    5400
.LSAQ      393     397     404     408     415     419     426     430     437     441     448     452     459     463        SEQ 1068
           470     474     478     482     486     490     494     501     505     512     516     523     527     534
           538     545     549     556     560     567     571     578     582     586     590     594     598     602
           609     613     617     621     625     629     633    2263    2265    2267    2269    2271    2273    2275
          2277    2363    2365    2367    2369    2479    2483    2487    2491    2495    2497    2499    2501    2503
          2505    2507    2509    2511    2515    2519    2523    2527    2531    2535    2539    2543    2599    2601
          2603    2605    2615    2715    2719    2723    2727    2790    2794    2798    2802    2841    2843    2845
          2861    2963    2967    2971    2975    3012    3014    3016    3018    3020    3022    3024    3026    3066
          3109    3113    3117    3121    3250    3252    3254    3256    3258    3331    3335    3420    3422    3424
          3426    3428    3430    3432    3434    3436    3438    3440    3442    3444    3446    3448    3450    3484
          3486    3488    3490    3492    3494    3496    3498    3500    3502    3504    3506    3508    3510    3512
          3514    3757    3823    4088    4154    4412    4472    4490    4740    4780    4798    5036    5096    5114
          5351    5391    5409
.LSDO      198     209     220     231     242     253     264     275     277     279     281     283     285     287
           739     743     829     833     919     923    1018    1022    1117    1121    1216    1220    1315    1319
          1416    1420    1577    1593    1609    1625    1641    1657    1673    1689    1705    1721    1737    1753
          1769    1785    1801    1817    1912    2051    2053    2055    2057    2059    2061    2063    2065    2067
          2069    2071    2073    2075    2077    2079    2081    2083    2245    2318    2357    2475    2550    2593
          2709    2711    2713    2784    2835    2955    2957    3008    3010    3071    3073    3243    3325    3327
          3384
.LSELE     198     200     202     204     209     211     213     215     220     222     224     226     231     233
           235     237     242     244     246     248     253     255     257     259     264     266     268     270
           275     277     279     281     283     285     287     393     397     404     408     415     419     426
           430     437     441     448     452     459     463     470     474     478     482     486     490     494
           501     505     512     516     523     527     534     538     545     549     556     560     567     571
           578     582     586     590     594     598     602     609     613     617     621     625     629     633
           741     745     831     835     921     925    1020    1024    1119    1123    1218    1222    1317    1321
          1418    1422    1581    1588    1597    1604    1613    1620    1629    1636    1645    1652    1661    1668
          1677    1684    1693    1700    1709    1716    1725    1732    1741    1748    1757    1764    1773    1780
          1789    1796    1805    1812    1821    1828    1916    1925    2088    2090    2092    2094    2096    2098
          2100    2102    2104    2106    2108    2110    2112    2114    2116    2118    2120    2311    2313    2350
          2352    2379    2381    2545    2588    2615    2779    2830    2861    3003    3066    3149    3320    3379
          3516    3742    3761    3791    3812    3825    3834    3843    4073    4092    4122    4143    4156    4165
          4174    4397    4414    4442    4461    4474    4483    4492    4725    4742    4760    4769    4782    4791
          4800    5021    5038    5066    5085    5098    5107    5116    5336    5353    5371    5380    5393    5402
          5411
.LSKCN     739     743     829     833     919     923    1018    1022    1117    1121    1216    1220    1315    1319
          1416    1420    1579    1586    1595    1602    1611    1618    1627    1634    1643    1650    1659    1666
          1675    1682    1691    1698    1707    1714    1723    1730    1739    1746    1755    1762    1771    1778
          1787    1794    1803    1810    1819    1826    1914    1923    2051    2053    2055    2057    2059    2061
          2063    2065    2067    2069    2071    2073    2075    2077    2079    2081    2083
.LSMIN    1416    1420    5011    5013    5015    5019    5034    5036    5054    5060    5064    5075    5079    5083
          5096    5105    5114    5330    5334    5349    5351    5369    5378    5391    5400    5409
.LXNOR     391     395     402     406     413     417     424     428     435     439     446     450     457     461
           468     472     476     480     484     488     492    1117    1121    1584    1600    1616    1632    1648
          1664    1680    1696    1712    1728    1744    1760    1776    1792    1808    1824    1921    2320    2559
          2552    2554    2595    2597    2786    2788    2837    2839    2959    2961    3075    3077    3079    3245
          3247    3250    3252    3254    3256    3258    3260    3262    3264    3266    3268    3270    3274    3278
          3282    3286    3288    3290    3292    3294    3298    3304    3310    3316    3329    3331    3335    3341
          3347    3351    3355    3361    3367    3371    3375    3386    3420    3422    3424    3426    3428    3430
          3432    3434    3436    3438    3440    3442    3444    3446    3448    3450    3452    3454    3456    3458
          3460    3462    3464    3466    3468    3470    3472    3474    3476    3478    3480    3482    3749    3751
          3753  3796  3817  3839  4080  4082  4084  4127  4148  4170  4404  4406  4408  4447      SEQ 1069
          4449  4466  4488  4732  4734  4736  4765  4774  4796  5028  5030  5032  5071  5073
          5090  5112  5343  5345  5347  5376  5385  5407
.LXOR     1018  1022  2963  2967  2971  2975  2981  2987  2993  2999  3012  3014  3016  3018
          3020  3022  3024  3026  3028  3030  3032  3034  3036  3038  3040  3042  3044  3046
          3048  3050  3052  3054  3056  3058  3060  3062  3064  3081  3085  3089  3093  3099
          3105  3109  3113  3117  3121  3127  3133  3139  3145
.MA       1577  1579  1581  1584  1586  1588  1593  1595  1597  1600  1602  1604  1609  1611
          1613  1616  1618  1620  1625  1627  1629  1632  1634  1636  1641  1643  1645  1648
          1650  1652  1657  1659  1661  1664  1666  1668  1673  1675  1677  1680  1682  1684
          1689  1691  1693  1696  1698  1700  1705  1707  1709  1712  1714  1716  1721  1723
          1725  1728  1730  1732  1737  1739  1741  1744  1746  1748  1753  1755  1757  1760
          1762  1764  1769  1771  1773  1776  1778  1780  1785  1787  1789  1792  1794  1796
          1801  1803  1805  1808  1810  1812  1817  1819  1821  1824  1826  1828  2051  2053
          2055  2057  2059  2061  2063  2065  2067  2069  2071  2073  2075  2077  2079  2081
          2083  2088  2090  2092  2094  2096  2098  2100  2102  2104  2106  2108  2110  2112
          2114  2116  2118  2120  2479  2483  2487  2491  2495  2499  2503  2507  2511  2515
          2519  2523  2527  2531  2535  2539  2543  2552  2554  2558  2562  2566  2570  2574
          2578  2582  2586  2588  2717  2721  2725  2729  2733  2737  2741  2745  2749  2753
          2757  2761  2765  2769  2773  2777  2779  2792  2796  2800  2804  2810  2816  2822
          2828  2830  2857  2859  2965  2969  2973  2977  2983  2989  2995  3001  3003  3083
          3087  3089  3091  3093  3095  3097  3101  3103  3107  3109  3111  3113  3115  3117
          3119  3121  3123  3125  3129  3131  3135  3137  3141  3143  3147  3260  3262  3264
          3266  3268  3272  3276  3280  3284  3286  3288  3290  3292  3294  3296  3300  3302
          3306  3308  3312  3314  3318  3333  3337  3343  3349  3353  3357  3363  3369  3373
          3377  3379  3420  3422  3424  3426  3428  3430  3432  3434  3436  3438  3440  3442
          3444  3446  3448  3450  3452  3454  3456  3458  3460  3462  3464  3466  3468  3470
          3472  3474  3476  3478  3480  3482  3484  3486  3488  3490  3492  3494  3496  3498
          3500  3502  3504  3506  3508  3510  3512  3514  3732  3738  3742  3755  3757  3759
          3761  3779  3781  3783  3785  3787  3798  3800  3802  3804  3806  3808  3823  3825
          3834  3843  4063  4069  4073  4086  4088  4090  4092  4110  4112  4114  4116  4118
          4129  4131  4133  4135  4137  4139  4154  4156  4165  4174  4397  4410  4412  4414
          4438  4442  4451  4453  4457  4461  4472  4474  4483  4492  4725  4738  4740  4742
          4760  4767  4769  4780  4782  4791  4800  5021  5034  5036  5038  5062  5066  5075
          5077  5081  5085  5096  5098  5107  5116  5336  5349  5351  5353  5371  5378  5380
          5391  5393  5402  5411
.MAND      198   200   202   204   209   211   213   215   220   222   224   226   231   233
           235   237   242   244   246   248   253   255   257   259   264   266   268   270
           829   833  1577  1593  1609  1625  1641  1657  1673  1689  1705  1721  1737  1753
          1769  1785  1801  1817  1912  1919  2245  2318  2357  2475  2477  2481  2485  2489
          2493  2497  2501  2505  2509  2513  2517  2521  2525  2529  2533  2537  2541  2550
          2556  2560  2564  2568  2572  2576  2580  2584  2593  2599  2601  2603  2605  2607
          2609  2611  2613  2615  2709  2711  2713  2784  2835  2861  2955  2957  3008  3010
          3071  3073  3083  3087  3091  3095  3101  3107  3111  3115  3119  3123  3129  3135
          3141  3147  3243  3325  3327  3333  3337  3339  3343  3345  3349  3353  3357  3359
          3363  3365  3369  3373  3377  3384  3484  3486  3488  3490  3492  3494  3496  3498
          3500  3502  3504  3506  3508  3510  3512  3514  3724  3726  3734  3747  3759  3766
          3819  4055  4057  4065  4078  4090  4097  4150  4385  4393  4402  4419  4468  4717
          4721  4730  4747  4776  5009  5017  5026  5043  5092  5328  5332  5341  5358  5387
.MB       1577  1579  1584  1586  1593  1595  1600  1602  1609  1611  1616  1618  1625  1627
          1632  1634  1641  1643  1648  1650  1657  1659  1664  1666  1673  1675  1680  1682
          1689  1691  1696  1698  1705  1707  1712  1714  1721  1723  1728  1730  1737  1739
          1744  1746  1753  1755  1760  1762  1769  1771  1776  1778  1785  1787  1792  1794
```

```
        1801  1803  1808  1810  1817  1819  1824  1826  2051  2053  2055  2057  2059  2061
        2063  2065  2067  2069  2071  2073  2075  2077  2079  2081  2083  2475  2481  2485
        2489  2493  2497  2501  2505  2509  2513  2517  2521  2525  2550  2556  2558  2560
        2562  2564  2566  2568  2570  2572  2574  2576  2578  2580  2582  2584  2586  2711
        2715  2717  2719  2721  2723  2725  2727  2729  2731  2733  2735  2737  2739  2741
        2743  2745  2747  2749  2751  2753  2755  2757  2759  2761  2763  2765  2767  2769
        2771  2773  2775  2777  2790  2792  2794  2796  2798  2800  2802  2804  2806  2808
        2810  2812  2814  2816  2818  2820  2822  2824  2826  2828  2839  2847  2849  2853
        2855  2857  2859  2957  2963  2965  2967  2969  2971  2973  2975  2977  2979  2981
        2983  2985  2987  2989  2991  2993  2995  2997  2999  3001  3038  3040  3042  3044
        3073  3077  3079  3081  3083  3085  3087  3089  3091  3093  3095  3097  3099  3101
        3103  3105  3107  3109  3111  3113  3115  3117  3119  3121  3123  3125  3127  3129
        3131  3133  3135  3137  3139  3141  3143  3145  3147  3245  3270  3274  3278  3282
        3296  3298  3302  3304  3308  3310  3314  3316  3329  3331  3333  3335  3337  3339
        3341  3343  3345  3347  3349  3351  3353  3355  3357  3359  3361  3363  3365  3367
        3369  3371  3373  3375  3377  3388  3390  3392  3394  3396  3398  3400  3402  3404
        3406  3408  3410  3412  3414  3416  3420  3422  3426  3428  3432  3434  3438  3440
        3444  3446  3452  3454  3456  3458  3460  3462  3464  3466  3468  3470  3472  3474
        3476  3478  3480  3482  3726  3728  3730  3732  3736  3738  3740  3749  3751  3755
        3757  3759  3777  3779  3781  3783  3785  3787  3796  3798  3800  3802  3804  3806
        3808  3817  3823  3830  3832  3839  3841  4057  4059  4061  4063  4067  4069  4071
        4080  4082  4086  4088  4090  4108  4110  4112  4114  4116  4118  4127  4129  4131
        4133  4135  4137  4139  4148  4154  4161  4163  4170  4172  4387  4389  4391  4395
        4397  4404  4406  4410  4412  4414  4430  4432  4436  4438  4440  4442  4447  4451
        4453  4455  4457  4459  4461  4468  4470  4472  4479  4481  4490  4719  4723  4725
        4732  4734  4738  4740  4742  4758  4765  4767  4776  4778  4780  4787  4789  4798
        5011  5013  5015  5019  5021  5028  5030  5034  5036  5038  5054  5056  5060  5062
        5064  5066  5071  5075  5077  5079  5081  5083  5085  5092  5094  5096  5103  5105
        5114  5330  5334  5336  5343  5345  5349  5351  5353  5369  5376  5378  5387  5389
        5391  5398  5400  5409
.MCONT  1579  1586  1595  1602  1611  1618  1627  1634  1643  1650  1659  1666  1675  1682
        1691  1698  1707  1714  1723  1730  1739  1746  1755  1762  1771  1778  1787  1794
        1803  1810  1819  1826  1914  1921  1923  2053  2055  2057  2059  2061  2063  2065
        2067  2069  2071  2073  2075  2077  2079  2081  2083  2247  2249  2251  2253  2255
        2257  2259  2261  2263  2265  2267  2269  2271  2273  2275  2277  2279  2281  2283
        2285  2287  2289  2291  2293  2295  2297  2299  2301  2303  2305  2307  2309  2320
        2322  2324  2326  2328  2330  2332  2334  2336  2338  2340  2342  2344  2346  2348
        2359  2361  2363  2365  2367  2369  2371  2373  2375  2377  2477  2479  2481  2483
        2485  2487  2489  2491  2493  2495  2497  2499  2501  2503  2505  2507  2509  2511
        2513  2515  2517  2519  2521  2523  2525  2527  2529  2531  2533  2535  2537  2539
        2541  2543  2552  2554  2556  2558  2560  2562  2564  2566  2568  2570  2572  2574
        2576  2578  2580  2582  2584  2586  2595  2597  2599  2601  2603  2605  2607  2609
        2611  2613  2711  2713  2715  2717  2719  2721  2723  2725  2727  2729  2731  2733
        2735  2737  2739  2741  2743  2745  2747  2749  2751  2753  2755  2757  2759  2761
        2763  2765  2767  2769  2771  2773  2775  2777  2786  2788  2790  2792  2794  2796
        2798  2800  2802  2804  2806  2808  2810  2812  2814  2816  2818  2820  2822  2824
        2826  2828  2837  2839  2841  2843  2845  2847  2849  2851  2853  2855  2857  2659
        2957  2959  2961  2963  2965  2967  2969  2971  2973  2975  2977  2979  2981  2983
        2985  2987  2989  2991  2993  2995  2997  2999  3001  3010  3012  3014  3016  3018
        3020  3022  3024  3026  3028  3030  3032  3034  3036  3038  3040  3042  3044  3046
        3048  3050  3052  3054  3056  3058  3060  3062  3064  3073  3075  3077  3079  3081
        3083  3085  3087  3089  3091  3093  3095  3097  3099  3101  3103  3105  3107  3109
        3111  3113  3115  3117  3119  3121  3123  3125  3127  3129  3131  3133  3135  3137
```

```
          3139  3141  3143  3145  3147  3245  3247  3250  3252  3254  3256  3258  3260  3262     SEQ 1071
          3264  3266  3268  3270  3272  3274  3276  3278  3280  3282  3284  3286  3288  3290
          3292  3294  3296  3298  3300  3302  3304  3306  3308  3310  3312  3314  3316  3318
          3327  3329  3331  3333  3335  3337  3339  3341  3343  3345  3347  3349  3351  3353
          3355  3357  3359  3361  3363  3365  3367  3369  3371  3373  3375  3377  3386  3388
          3390  3392  3394  3396  3398  3400  3402  3404  3406  3408  3410  3412  3414  3416
          3418  3420  3422  3424  3426  3428  3430  3432  3434  3436  3438  3440  3442  3444
          3446  3448  3450  3452  3454  3456  3458  3460  3462  3464  3466  3468  3470  3472
          3474  3476  3478  3480  3482  3484  3486  3488  3490  3492  3494  3496  3498  3500
          3502  3504  3506  3508  3510  3512  3514  3726  3728  3730  3732  3734  3736  3738
          3740  3749  3751  3753  3755  3757  3759  3772  3777  3779  3781  3783  3785  3787
          3789  3798  3800  3802  3804  3806  3808  3810  3819  3821  3823  3832  3841  4057
          4059  4061  4063  4065  4067  4069  4071  4080  4082  4084  4086  4088  4090  4103
          4108  4110  4112  4114  4116  4118  4120  4129  4131  4133  4135  4137  4139  4141
          4150  4152  4154  4163  4172  4387  4389  4391  4393  4395  4404  4406  4408  4410
          4412  4425  4430  4432  4434  4436  4438  4440  4449  4451  4453  4455  4457  4459
          4468  4470  4472  4481  4490  4719  4721  4723  4732  4734  4736  4738  4740  4753
          4758  4767  4776  4778  4780  4789  4798  5011  5013  5015  5017  5019  5028  5030
          5032  5034  5036  5049  5054  5056  5058  5060  5062  5064  5073  5075  5077  5079
          5081  5083  5092  5094  5096  5105  5114  5330  5332  5334  5343  5345  5347  5349
          5351  5364  5369  5378  5387  5389  5391  5400  5409
.MCRY     1315  1319  2263  2265  2267  2269  2271  2273  2275  2277  2291  2303  2330  2334
          2336  2340  2342  2363  2367  2373  2489  2493  2497  2501  2517  2521  2533  2541
          2556  2560  2564  2568  2580  2584  2601  2603  2605  2607  2715  2719  2723  2727
          2739  2747  2759  2767  2802  2808  2826  2843  2845  2847  2849  2967  2971  2975
          2981  3014  3016  3018  3022  3032  3042  3050  3052  3062  3081  3085  3089  3093
          3105  3117  3121  3133  3139  3250  3252  3254  3256  3258  3288  3290  3316  3335
          3337  3339  3341  3343  3345  3347  3768  3821  4059  4061  4067  4071  4086  4088
          4099  4108  4110  4112  4114  4116  4118  4120  4129  4131  4133  4135  4137  4139
          4141  4152  4154  4163  4172  4421  4470  4719  4723  4738  4740  4749  4758  4767
          4778  4780  4789  4798  5045  5094  5330  5334  5349  5351  5360  5369  5378  5389
          5391  5400  5409
.MD        198   200   202   204   209   211   213   215   220   222   224   226   231   233
           235   237   242   244   246   248   253   255   257   259   264   266   268   270
           275   277   279   281   283   285   287   391   393   395   397   402   404   406
           408   413   415   417   419   424   426   428   430   435   437   439   441   446
           448   450   452   457   459   461   463   468   470   472   474   476   478   480
           482   484   486   488   490   492   494   499   501   503   505   510   512   514
           516   521   523   525   527   532   534   536   538   543   545   547   549   554
           556   558   560   565   567   569   571   576   578   580   582   584   586   588
           590   592   594   596   598   600   602   607   609   611   613   615   617   619
           621   623   625   627   629   631   633   739   741   743   745   829   831   833
           835   919   921   923   925  1018  1020  1022  1024  1117  1119  1121  1123  1216
          1218  1220  1222  1315  1317  1319  1321  1416  1418  1420  1422  1577  1579  1581
          1584  1586  1588  1593  1595  1597  1600  1602  1604  1609  1611  1613  1616  1618
          1620  1625  1627  1629  1632  1634  1636  1641  1643  1645  1648  1650  1652  1657
          1659  1661  1664  1666  1668  1673  1675  1677  1680  1682  1684  1689  1691  1693
          1696  1698  1700  1705  1707  1709  1712  1714  1716  1721  1723  1725  1728  1730
          1732  1737  1739  1741  1744  1746  1748  1753  1755  1757  1760  1762  1764  1769
          1771  1773  1776  1778  1780  1785  1787  1789  1792  1794  1796  1801  1803  1805
          1808  1810  1812  1817  1819  1821  1824  1826  1828  1912  1914  1916  1919  1921
          1923  1925  2051  2053  2055  2057  2059  2061  2063  2065  2067  2069  2071  2073
          2075  2077  2079  2081  2083  2088  2090  2092  2094  2096  2098  2100  2102  2104
          2106  2108  2110  2112  2114  2116  2118  2120  2245  2247  2249  2251  2253  2255     SEQ 1072
          2257  2259  2261  2263  2265  2267  2269  2271  2273  2275  2277  2279  2281  2283
          2285  2287  2289  2291  2293  2295  2297  2299  2301  2303  2305  2307  2309  2311
          2313  2318  2320  2322  2324  2326  2328  2330  2332  2334  2336  2338  2340  2342
          2344  2346  2348  2350  2352  2357  2359  2361  2363  2365  2367  2369  2371  2373
          2375  2377  2379  2381  2475  2477  2479  2481  2483  2485  2487  2489  2491  2493
          2495  2497  2499  2501  2503  2505  2507  2509  2511  2513  2515  2517  2519  2521
          2523  2525  2527  2529  2531  2533  2535  2537  2539  2541  2543  2545  2550  2552
          2554  2556  2558  2560  2562  2564  2566  2568  2570  2572  2574  2576  2578  2580
          2582  2584  2586  2588  2593  2595  2597  2599  2601  2603  2605  2607  2609  2611
          2613  2615  2709  2711  2713  2715  2717  2719  2721  2723  2725  2727  2729  2731
          2733  2735  2737  2739  2741  2743  2745  2747  2749  2751  2753  2755  2757  2759
          2761  2763  2765  2767  2769  2771  2773  2775  2777  2779  2784  2786  2788  2790
          2792  2794  2796  2798  2800  2802  2804  2806  2808  2810  2812  2814  2816  2818
          2820  2822  2824  2826  2828  2830  2835  2837  2839  2841  2843  2845  2847  2849
          2851  2853  2855  2857  2859  2861  2955  2957  2959  2961  2963  2965  2967  2969
          2971  2973  2975  2977  2979  2981  2983  2985  2987  2989  2991  2993  2995  2997
          2999  3001  3003  3008  3010  3012  3014  3016  3018  3020  3022  3024  3026  3028
          3030  3032  3034  3036  3038  3040  3042  3044  3046  3048  3050  3052  3054  3056
          3058  3060  3062  3064  3066  3071  3073  3075  3077  3079  3081  3083  3085  3087
          3089  3091  3093  3095  3097  3099  3101  3103  3105  3107  3109  3111  3113  3115
          3117  3119  3121  3123  3125  3127  3129  3131  3133  3135  3137  3139  3141  3143
          3145  3147  3149  3243  3245  3247  3250  3252  3254  3256  3258  3260  3262  3264
          3266  3268  3270  3272  3274  3276  3278  3280  3282  3284  3286  3288  3290  3292
          3294  3296  3298  3300  3302  3304  3306  3308  3310  3312  3314  3316  3318  3320
          3325  3327  3329  3331  3333  3335  3337  3339  3341  3343  3345  3347  3349  3351
          3353  3355  3357  3359  3361  3363  3365  3367  3369  3371  3373  3375  3377  3379
          3384  3386  3388  3390  3392  3394  3396  3398  3400  3402  3404  3406  3408  3410
          3412  3414  3416  3418  3420  3422  3424  3426  3428  3430  3432  3434  3436  3438
          3440  3442  3444  3446  3448  3450  3452  3454  3456  3458  3460  3462  3464  3466
          3468  3470  3472  3474  3476  3478  3480  3482  3484  3486  3488  3490  3492  3494
          3496  3498  3500  3502  3504  3506  3508  3510  3512  3514  3516  3724  3726  3728
          3730  3732  3734  3736  3738  3740  3742  3747  3749  3751  3753  3755  3757  3759
          3761  3766  3768  3770  3772  3777  3779  3781  3783  3785  3787  3789  3791  3796
          3798  3800  3802  3804  3806  3808  3810  3812  3817  3819  3821  3823  3825  3830
          3832  3834  3839  3841  3843  4055  4057  4059  4061  4063  4065  4067  4069  4071
          4073  4078  4080  4082  4084  4086  4088  4090  4092  4097  4099  4101  4103  4108
          4110  4112  4114  4116  4118  4120  4122  4127  4129  4131  4133  4135  4137  4139
          4141  4143  4148  4150  4152  4154  4156  4161  4163  4165  4170  4172  4174  4385
          4387  4389  4391  4393  4395  4397  4402  4404  4406  4408  4410  4412  4414  4419
          4421  4423  4425  4430  4432  4434  4436  4438  4440  4442  4447  4449  4451  4453
          4455  4457  4459  4461  4466  4468  4470  4472  4474  4479  4481  4483  4488  4490
          4492  4717  4719  4721  4723  4725  4730  4732  4734  4736  4738  4740  4742  4747
          4749  4751  4753  4758  4760  4765  4767  4769  4774  4776  4778  4780  4782  4787
          4789  4791  4796  4798  4800  5009  5011  5013  5015  5017  5019  5021  5026  5028
          5030  5032  5034  5036  5038  5043  5045  5047  5049  5054  5056  5058  5060  5062
          5064  5066  5071  5073  5075  5077  5079  5081  5083  5085  5090  5092  5094  5096
          5098  5103  5105  5107  5112  5114  5116  5328  5330  5332  5334  5336  5341  5343
          5345  5347  5349  5351  5353  5358  5360  5362  5364  5369  5371  5376  5378  5380
          5385  5387  5389  5391  5393  5398  5400  5402  5407  5409  5411
.MJ        198   200   202   204   209   211   213   215   220   222   224   226   231   233
           235   237   242   244   246   248   253   255   257   259   264   266   268   270
           275   277   279   281   283   285   287   391   393   395   397   402   404   406
          408   413   415   417   419   424   426   428   430   435   437   439   441   446      SEQ 1073
          448   450   452   457   459   461   463   468   470   472   474   476   478   480
          482   484   486   488   490   492   494   499   501   503   505   510   512   514
          516   521   523   525   527   532   534   536   538   543   545   547   549   554
          556   558   560   565   567   569   571   576   578   580   582   584   586   588
          590   592   594   596   598   600   602   607   609   611   613   615   617   619
          621   623   625   627   629   631   633   739   741   743   745   829   831   833
          835   919   921   923   925  1018  1020  1022  1024  1117  1119  1121  1123  1216
         1218  1220  1222  1315  1317  1319  1321  1416  1418  1420  1422  1577  1581  1584
         1588  1593  1597  1600  1604  1609  1613  1616  1620  1625  1629  1632  1636  1641
         1645  1648  1652  1657  1661  1664  1668  1673  1677  1680  1684  1689  1693  1696
         1700  1705  1709  1712  1716  1721  1725  1728  1732  1737  1741  1744  1748  1753
         1757  1760  1764  1769  1773  1776  1780  1785  1789  1792  1796  1801  1805  1808
         1812  1817  1821  1824  1828  1912  1916  1919  1925  2051  2088  2090  2092  2094
         2096  2098  2100  2102  2104  2106  2108  2110  2112  2114  2116  2118  2120  2245
         2311  2313  2318  2350  2352  2357  2379  2381  2475  2545  2550  2588  2593  2615
         2709  2779  2784  2830  2835  2861  2955  3003  3008  3066  3071  3149  3243  3320
         3325  3379  3384  3516  3724  3742  3747  3761  3766  3768  3770  3791  3796  3812
         3817  3825  3830  3834  3839  3843  4055  4073  4078  4092  4097  4099  4101  4122
         4127  4143  4148  4156  4161  4165  4170  4174  4385  4397  4402  4414  4419  4421
         4423  4442  4447  4461  4466  4474  4479  4483  4488  4492  4717  4725  4730  4742
         4747  4749  4751  4760  4765  4769  4774  4782  4787  4791  4796  4800  5009  5021
         5026  5038  5043  5045  5047  5066  5071  5085  5090  5098  5103  5107  5112  5116
         5328  5336  5341  5353  5358  5360  5362  5371  5376  5380  5385  5393  5398  5402
         5407  5411
.MJMAP    198   200   202   204   209   211   213   215   220   222   224   226   231   233
          235   237   242   244   246   248   253   255   257   259   264   266   268   270
          275   277   279   281   283   285   287   391   393   395   397   402   404   406
          408   413   415   417   419   424   426   428   430   435   437   439   441   446
          448   450   452   457   459   461   463   468   470   472   474   476   478   480
          482   484   486   488   490   492   494   499   501   503   505   510   512   514
          516   521   523   525   527   532   534   536   538   543   545   547   549   554
          556   558   560   565   567   569   571   576   578   580   582   584   586   588
          590   592   594   596   598   600   602   607   609   611   613   615   617   619
          621   623   625   627   629   631   633   739   741   743   745   829   831   833
          835   919   921   923   925  1018  1020  1022  1024  1117  1119  1121  1123  1216
         1218  1220  1222  1315  1317  1319  1321  1416  1418  1420  1422  1577  1581  1584
         1588  1593  1597  1600  1604  1609  1613  1616  1620  1625  1629  1632  1636  1641
         1645  1648  1652  1657  1661  1664  1668  1673  1677  1680  1684  1689  1693  1696
         1700  1705  1709  1712  1716  1721  1725  1728  1732  1737  1741  1744  1748  1753
         1757  1760  1764  1769  1773  1776  1780  1785  1789  1792  1796  1801  1805  1808
         1812  1817  1821  1824  1828  1912  1916  1919  1925  2051  2088  2090  2092  2094
         2096  2098  2100  2102  2104  2106  2108  2110  2112  2114  2116  2118  2120  2245
         2311  2313  2318  2350  2352  2357  2379  2381  2475  2545  2550  2588  2593  2615
         2709  2779  2784  2830  2835  2861  2955  3003  3008  3066  3071  3149  3243  3320
         3325  3379  3384  3516  3724  3742  3747  3761  3766  3768  3770  3791  3796  3812
         3817  3825  3830  3834  3839  3843  4055  4073  4078  4092  4097  4099  4101  4122
         4127  4143  4148  4156  4161  4165  4170  4174  4385  4397  4402  4414  4419  4421
         4423  4442  4447  4461  4466  4474  4479  4483  4488  4492  4717  4725  4730  4742
         4747  4749  4751  4760  4765  4769  4774  4782  4787  4791  4796  4800  5009  5021
         5026  5038  5043  5045  5047  5066  5071  5085  5090  5098  5103  5107  5112  5116
         5328  5336  5341  5353  5358  5360  5362  5371  5376  5380  5385  5393  5398  5402
         5407  5411
```

```
.MMGC      198    200    202    204    209    211    213    215    220    222    224    226    231    233     SEQ 1074
           235    237    242    244    246    248    253    255    257    259    264    266    268    270
           275    277    279    281    283    285    287    393    397    404    408    415    419    426
           430    437    441    448    452    459    463    470    474    478    482    486    490    494
           501    505    512    516    523    527    534    538    545    549    556    560    567    571
           578    582    586    590    594    598    602    609    613    617    621    625    629    633
           739    741    743    745    829    831    833    835    919    921    923    925   1018   1020
          1022   1024   1117   1119   1121   1123   1216   1218   1220   1222   1315   1317   1319   1321
          1416   1418   1420   1422   1579   1581   1586   1588   1595   1597   1602   1604   1611   1613
          1618   1620   1627   1629   1634   1636   1643   1645   1650   1652   1659   1661   1666   1668
          1675   1677   1682   1684   1691   1693   1698   1700   1707   1709   1714   1716   1723   1725
          1730   1732   1739   1741   1746   1748   1755   1757   1762   1764   1771   1773   1778   1780
          1787   1789   1794   1796   1803   1805   1810   1812   1819   1821   1826   1828   1914   1916
          1923   1925   2051   2053   2055   2057   2059   2061   2063   2065   2067   2069   2071   2073
          2075   2077   2079   2081   2083   2088   2090   2092   2094   2096   2098   2100   2102   2104
          2106   2108   2110   2112   2114   2116   2118   2120   2311   2313   2350   2352   2379   2381
          2545   2588   2615   2779   2830   2861   3003   3066   3149   3320   3379   3516   3742   3761
          3791   3812   3825   3834   3843   4073   4092   4122   4143   4156   4165   4174   4397   4414
          4442   4461   4474   4483   4492   4725   4742   4760   4769   4782   4791   4800   5021   5038
          5066   5085   5098   5107   5116   5336   5353   5371   5380   5393   5402   5411
.MNAND     275    277    279    281    283    285    287    919    923   2715   2719   2723   2727   2731
          2735   2739   2743   2747   2751   2755   2759   2763   2767   2771   2775   2790   2794   2798
          2802   2808   2814   2820   2826   2841   2843   2845   2847   2849   2851   2853   2855   2857
          2859
.MOENA     198    200    202    204    209    211    213    215    220    222    224    226    231    233
           235    237    242    244    246    248    253    255    257    259    264    266    268    270
           275    277    279    281    283    285    287    393    397    404    408    415    419    426
           430    437    441    448    452    459    463    470    474    478    482    486    490    494
           501    505    512    516    523    527    534    538    545    549    556    560    567    571
           578    582    586    590    594    598    602    609    613    617    621    625    629    633
           741    745    831    835    921    925   1020   1024   1119   1123   1218   1222   1317   1321
          1418   1422   1581   1588   1597   1604   1613   1620   1629   1636   1645   1652   1661   1668
          1677   1684   1693   1700   1709   1716   1725   1732   1741   1748   1757   1764   1773   1780
          1789   1796   1805   1812   1821   1828   1916   1925   2088   2090   2092   2094   2096   2098
          2100   2102   2104   2106   2108   2110   2112   2114   2116   2118   2120   2311   2313   2350
          2352   2379   2381   2545   2588   2615   2779   2830   2861   3003   3066   3149   3320   3379
          3516   3742   3761   3791   3812   3825   3834   3843   4073   4092   4122   4143   4156   4165
          4174   4397   4414   4442   4461   4474   4483   4492   4725   4742   4760   4769   4782   4791
          4800   5021   5038   5066   5085   5098   5107   5116   5336   5353   5371   5380   5393   5402
          5411
.MOR       393    397    404    408    415    419    426    430    437    441    448    452    459    463
           470    474    478    482    486    490    494    501    505    512    516    523    527    534
           538    545    549    556    560    567    571    578    582    586    590    594    598    602
           609    613    617    621    625    629    633    739    741    743    745    831    835    921
           925   1020   1024   1119   1123   1218   1222   1317   1321   1418   1422   1579   1581   1586
          1588   1595   1597   1602   1604   1611   1613   1618   1620   1627   1629   1634   1636   1643
          1645   1650   1652   1659   1661   1666   1668   1675   1677   1682   1684   1691   1693   1698
          1700   1707   1709   1714   1716   1723   1725   1730   1732   1739   1741   1746   1748   1755
          1757   1762   1764   1771   1773   1778   1780   1787   1789   1794   1796   1803   1805   1810
          1812   1819   1821   1826   1828   1914   1916   1923   1925   2051   2053   2055   2057   2059
          2061   2063   2065   2067   2069   2071   2073   2075   2077   2079   2081   2083   2088   2090
          2092   2094   2096   2098   2100   2102   2104   2106   2108   2110   2112   2114   2116   2118
          2120   2247   2249   2251   2253   2255   2257   2259   2261   2263   2265   2267   2269   2271
```

```
              2273     2275     2277     2279     2281     2283     2285     2287     2289     2291     2293     2295     2297     2299     SEQ 1075
              2301     2303     2305     2307     2309     2311     2313     2322     2324     2326     2328     2330     2332     2334
              2336     2338     2340     2342     2344     2346     2348     2350     2352     2361     2363     2365     2367     2369
              2371     2373     2375     2377     2379     2381     2479     2483     2487     2491     2495     2499     2503     2507
              2511     2515     2519     2523     2527     2531     2535     2539     2543     2545     2558     2562     2566     2570
              2574     2578     2582     2586     2588     2717     2721     2725     2729     2733     2737     2741     2745     2749
              2753     2757     2761     2765     2769     2773     2777     2779     2792     2796     2800     2804     2806     2810
              2812     2816     2818     2822     2824     2828     2830     2965     2969     2973     2977     2979     2983     2985
              2989     2991     2995     2997     3001     3003     3066     3097     3103     3125     3131     3137     3143     3149
              3272     3276     3280     3284     3296     3300     3302     3306     3308     3312     3314     3318     3320     3379
              3388     3390     3392     3394     3396     3398     3400     3402     3404     3406     3408     3410     3412     3414
              3416     3418     3516     3732     3738     3742     3761     3770     3772     3791     3812     3825     3830     3834
              3843     4073     4092     4101     4103     4122     4143     4156     4161     4165     4174     4397     4414     4423
              4425     4432     4434     4438     4442     4453     4457     4461     4474     4479     4483     4492     4751     4753
              4760     4769     4782     4787     4791     4800     5021     5038     5047     5049     5056     5058     5062     5066
              5077     5081     5085     5098     5103     5107     5116     5362     5364     5371     5380     5393     5398     5402
              5411
.MPLUS        1216     1220     3728     3730     3736     3740     3755     3757     3768     3777     3779     3781     3783     3785
              3787     3789     3798     3800     3802     3804     3806     3808     3810     3821     3823     3832     3841     4059
              4061     4063     4067     4069     4071     4086     4088     4099     4108     4110     4112     4114     4116     4118
              4120     4129     4131     4133     4135     4137     4139     4141     4152     4154     4163     4172     4421     4470
              4725     4742     4749     4778     5045     5094     5336     5353     5360     5389
.MRMIN         499      503      510      514      521      525      532      536      543      547      554      558      565      569
               576      580      584      588      592      596      600      607      611      615      619      623      627      631
              1315     1319     4387     4389     4391     4395     4410     4412     4430     4436     4440     4451     4455     4459
              4472     4481     4490     4719     4723     4738     4740     4758     4767     4780     4789     4798
.MSOA          204      215      226      237      248      259      270      468      472      476      480      484      488      492
               576      580      584      588      592      596      600     1581     1584     1588     1597     1600     1604     1613
              1616     1620     1629     1632     1636     1645     1648     1652     1661     1664     1668     1677     1680     1684
              1693     1696     1700     1709     1712     1716     1725     1728     1732     1741     1744     1748     1757     1760
              1764     1773     1776     1780     1789     1792     1796     1805     1808     1812     1821     1824     1828     1919
              1921     2088     2090     2092     2094     2096     2098     2100     2102     2104     2106     2108     2110     2112
              2114     2116     2118     2247     2249     2251     2253     2255     2257     2259     2261     2311     2320     2334
              2336     2338     2340     2350     2359     2379     2477     2481     2485     2489     2493     2552     2554     2572
              2576     2580     2584     2588     2595     2597     2763     2767     2771     2775     2779     2786     2788     2806
              2812     2818     2824     2830     2837     2839     2857     2859     2959     2961     2979     2985     2991     2997
              3003     3046     3048     3050     3052     3054     3075     3077     3079     3089     3093     3097     3103     3125
              3131     3137     3143     3245     3247     3286     3288     3290     3292     3294     3296     3302     3308     3314
              3320     3329     3371     3375     3379     3388     3390     3392     3394     3396     3398     3400     3402     3404
              3406     3408     3410     3412     3414     3416     3418     3516     3724     3726     3730     3734     3742     3747
              3749     3751     3753     3761     3766     3768     3770     3772     3777     3779     3781     3796     3798     3800
              3802     3817     3819     3825     3830     3834     3839     3843     4055     4057     4061     4065     4073     4078
              4080     4082     4084     4092     4097     4099     4101     4103     4108     4110     4112     4127     4129     4131
              4133     4148     4150     4156     4161     4165     4170     4174     4385     4387     4393     4402     4404     4406
              4408     4419     4421     4423     4425     4430     4432     4434     4447     4449     4451     4453     4466     4474
              4479     4483     4488     4492     4717     4719     4721     4730     4732     4734     4736     4747     4749     4751
              4753     4758     4760     4765     4767     4769     4774     4782     4787     4791     4796     4800     5009     5011
              5017     5026     5028     5030     5032     5043     5045     5047     5049     5054     5056     5058     5071     5073
              5075     5077     5090     5098     5103     5107     5112     5116     5328     5330     5332     5341     5343     5345
              5347     5358     5360     5362     5364     5369     5371     5376     5378     5380     5385     5393     5398     5402
              5407     5411
.MSOB          202      213      224      235      246      257      268      607      611      615      619      623      627      631
              2295     2297     2299     2301     2303     2305     2307     2309     2342     2344     2346     2348     2529     2533
```

```
            2537    2541    2853    2855    3056    3058    3060    3062    3064    3099    3105    3298    3304    3310        SEQ 1076
            3316    3339    3345    3359    3361    3365    3367    3740    3783    3785    3787    3804    3806    3808
            4071    4114    4116    4118    4135    4137    4139    4389    4436    4455    4468    4470    4776    4778
            5013    5060    5079    5092    5094    5387    5389
.MSOQ        200     211     222     233     244     255     266     391     395     402     406     413     417     424
             428     435     439     446     450     457     461     499     503     510     514     521     525     532
             536     543     547     554     558     565     569     741     745     831     835     921     925    1020
            1024    1119    1123    1218    1222    1317    1321    1418    1422    1914    1916    1923    1925    2120
            2279    2281    2283    2285    2287    2289    2291    2293    2313    2322    2324    2326    2328    2330
            2332    2352    2361    2381    2513    2517    2521    2525    2545    2556    2560    2564    2568    2747
            2751    2755    2759    2851    3038    3040    3042    3044    3081    3085    3149    3270    3274    3278
            3282    3351    3355    3736    3789    3791    3810    3812    3821    4067    4120    4122    4141    4143
            4152    4395    4440    4459    4723    5019    5064    5083    5334
.MSAB       1579    1586    1595    1602    1611    1618    1627    1634    1643    1650    1659    1666    1675    1682
            1691    1698    1707    1714    1723    1730    1739    1746    1755    1762    1771    1778    1787    1794
            1803    1810    1819    1826    2371    2373    2375    2377    2558    2562    2566    2570    2574    2578
            2582    2586    2607    2609    2611    2613    2717    2721    2725    2729    2731    2733    2735    2737
            2739    2741    2743    2745    2749    2753    2757    2761    2765    2769    2773    2777    2792    2796
            2800    2804    2808    2810    2814    2816    2820    2822    2826    2828    2847    2849    2965    2969
            2973    2977    2981    2983    2987    2989    2993    2995    2999    3001    3028    3030    3032    3034
            3036    3083    3087    3091    3095    3101    3107    3111    3115    3119    3123    3127    3129    3133
            3135    3139    3141    3145    3147    3260    3262    3264    3266    3268    3272    3276    3280    3284
            3300    3306    3312    3318    3333    3337    3341    3343    3347    3349    3353    3357    3363    3369
            3373    3377    3386    3452    3454    3456    3458    3460    3462    3464    3466    3468    3470    3472
            3474    3476    3478    3480    3482    3728    3732    3738    3755    3759    3832    3841    4059    4063
            4069    4086    4090    4163    4172    4391    4397    4410    4414    4438    4442    4457    4461    4481
            4725    4738    4742    4789    5015    5021    5034    5038    5062    5066    5081    5085    5105    5336
            5349    5353    5400
.MSAQ        393     397     404     408     415     419     426     430     437     441     448     452     459     463
             470     474     478     482     486     490     494     501     505     512     516     523     527     534
             538     545     549     556     560     567     571     578     582     586     590     594     598     602
             609     613     617     621     625     629     633    2263    2265    2267    2269    2271    2273    2275
            2277    2363    2365    2367    2369    2479    2483    2487    2491    2495    2497    2499    2501    2503
            2505    2507    2509    2511    2515    2519    2523    2527    2531    2535    2539    2543    2599    2601
            2603    2605    2615    2715    2719    2723    2727    2790    2794    2798    2802    2841    2843    2845
            2861    2963    2967    2971    2975    3012    3014    3016    3018    3020    3022    3024    3026    3066
            3109    3113    3117    3121    3250    3252    3254    3256    3258    3331    3335    3420    3422    3424
            3426    3428    3430    3432    3434    3436    3438    3440    3442    3444    3446    3448    3450    3484
            3486    3488    3490    3492    3494    3496    3498    3500    3502    3504    3506    3508    3510    3512
            3514    3757    3823    4088    4154    4412    4472    4490    4740    4780    4798    5036    5096    5114
            5351    5391    5409
.MSDO        198     209     220     231     242     253     264     275     277     279     281     283     285     287
             739     743     829     833     919     923    1018    1022    1117    1121    1216    1220    1315    1319
            1416    1420    1577    1593    1609    1625    1641    1657    1673    1689    1705    1721    1737    1753
            1769    1785    1801    1817    1912    2051    2053    2055    2057    2059    2061    2063    2065    2067
            2069    2071    2073    2075    2077    2079    2081    2083    2245    2318    2357    2475    2550    2593
            2709    2711    2713    2784    2835    2955    2957    3008    3010    3071    3073    3243    3325    3527
            3384
.MSELE       198     200     202     204     209     211     213     215     220     222     224     226     231     233
             235     237     242     244     246     248     253     255     257     259     264     266     268     270
             275     277     279     281     283     285     287     393     397     404     408     415     419     426
             430     437     441     448     452     459     463     470     474     478     482     486     490     494
             501     505     512     516     523     527     534     538     545     549     556     560     567     571
```

```
         578   582   586   590   594   598   602   609   613   617   621   625   629   633
         741   745   831   835   921   925  1020  1024  1119  1123  1218  1222  1317  1321
        1418  1422  1581  1588  1597  1604  1613  1620  1629  1636  1645  1652  1661  1668
        1677  1684  1693  1700  1709  1716  1725  1732  1741  1748  1757  1764  1773  1780
        1789  1796  1805  1812  1821  1828  1916  1925  2088  2090  2092  2094  2096  2098
        2100  2102  2104  2106  2108  2110  2112  2114  2116  2118  2120  2311  2313  2350
        2352  2379  2381  2545  2588  2615  2779  2830  2861  3003  3066  3149  3320  3379
        3516  3742  3761  3791  3812  3825  3834  3843  4073  4092  4122  4143  4156  4165
        4174  4397  4414  4442  4461  4474  4483  4492  4725  4742  4760  4769  4782  4791
        4800  5021  5038  5066  5085  5098  5107  5116  5336  5353  5371  5380  5393  5402
        5411
.MSKCN   739   743   829   833   919   923  1018  1022  1117  1121  1216  1220  1315  1319
        1416  1420  1579  1586  1595  1602  1611  1618  1627  1634  1643  1650  1659  1666
        1675  1682  1691  1698  1707  1714  1723  1730  1739  1746  1755  1762  1771  1778
        1787  1794  1803  1810  1819  1826  1914  1923  2051  2053  2055  2057  2059  2061
        2063  2065  2067  2069  2071  2073  2075  2077  2079  2081  2083
.MSMIN  1416  1420  5011  5013  5015  5019  5034  5036  5054  5060  5064  5075  5079  5083
        5096  5105  5114  5330  5334  5349  5351  5369  5378  5391  5400  5409
.MXNOR   391   395   402   406   413   417   424   428   435   439   446   450   457   461
         468   472   476   480   484   488   492  1117  1121  1584  1600  1616  1632  1648
        1664  1680  1696  1712  1728  1744  1760  1776  1792  1808  1824  1921  2320  2359
        2552  2554  2595  2597  2786  2788  2837  2839  2959  2961  3075  3077  3079  3245
        3247  3250  3252  3254  3256  3258  3260  3262  3264  3266  3268  3270  3274  3278
        3282  3286  3288  3290  3292  3294  3298  3304  3310  3316  3329  3331  3335  3341
        3347  3351  3355  3361  3367  3371  3375  3386  3420  3422  3424  3426  3428  3430
        3432  3434  3436  3438  3440  3442  3444  3446  3448  3450  3452  3454  3456  3458
        3460  3462  3464  3466  3468  3470  3472  3474  3476  3478  3480  3482  3749  3751
        3753  3796  3817  3839  4080  4082  4084  4127  4148  4170  4404  4406  4408  4447
        4449  4466  4488  4732  4734  4736  4765  4774  4796  5028  5030  5032  5071  5073
        5090  5112  5343  5345  5347  5376  5385  5407
.MXOR   1018  1022  2963  2967  2971  2975  2981  2987  2993  2999  3012  3014  3016  3018
        3020  3022  3024  3026  3028  3030  3032  3034  3036  3038  3040  3042  3044  3046
        3048  3050  3052  3054  3056  3058  3060  3062  3064  3081  3085  3089  3093  3099
        3105  3109  3113  3117  3121  3127  3133  3139  3145
.RA     1577  1579  1581  1584  1586  1588  1593  1595  1597  1600  1602  1604  1609  1611
        1613  1616  1618  1620  1625  1627  1629  1632  1634  1636  1641  1643  1645  1648
        1650  1652  1657  1659  1661  1664  1666  1668  1673  1675  1677  1680  1682  1684
        1689  1691  1693  1696  1698  1700  1705  1707  1709  1712  1714  1716  1721  1723
        1725  1728  1730  1732  1737  1739  1741  1744  1746  1748  1753  1755  1757  1760
        1762  1764  1769  1771  1773  1776  1778  1780  1785  1787  1789  1792  1794  1796
        1801  1803  1805  1808  1810  1812  1817  1819  1821  1824  1826  1828  2051  2053
        2055  2057  2059  2061  2063  2065  2067  2069  2071  2073  2075  2077  2079  2081
        2083  2088  2090  2092  2094  2096  2098  2100  2102  2104  2106  2108  2110  2112
        2114  2116  2118  2120  2479  2483  2487  2491  2495  2499  2503  2507  2511  2515
        2519  2523  2527  2531  2535  2539  2543  2552  2554  2558  2562  2566  2570  2574
        2578  2582  2586  2588  2717  2721  2725  2729  2733  2737  2741  2745  2749  2753
        2757  2761  2765  2769  2773  2777  2779  2792  2796  2800  2804  2810  2816  2622
        2828  2830  2857  2859  2965  2969  2973  2977  2983  2989  2995  3001  3003  3083
        3087  3089  3091  3093  3095  3097  3101  3103  3107  3109  3111  3113  3115  3117
        3119  3121  3123  3125  3129  3131  3135  3137  3141  3143  3147  3260  3262  3264
        3266  3268  3272  3276  3280  3284  3286  3288  3290  3292  3294  3296  3300  3302
        3306  3308  3312  3314  3318  3333  3337  3343  3349  3353  3357  3363  3369  3373
        3377  3379  3420  3422  3424  3426  3428  3430  3432  3434  3436  3438  3440  3442
```

```
          3444  3446  3448  3450  3452  3454  3456  3458  3460  3462  3464  3466  3468  3470
          3472  3474  3476  3478  3480  3482  3484  3486  3488  3490  3492  3494  3496  3498
          3500  3502  3504  3506  3508  3510  3512  3514  3732  3738  3742  3755  3757  3759
          3761  3779  3781  3783  3785  3787  3798  3800  3802  3804  3806  3808  3823  3825
          3834  3843  4063  4069  4073  4086  4088  4090  4092  4110  4112  4114  4116  4118
          4129  4131  4133  4135  4137  4139  4154  4156  4165  4174  4397  4410  4412  4414
          4438  4442  4451  4453  4457  4461  4472  4474  4483  4492  4725  4738  4740  4742
          4760  4767  4769  4780  4782  4791  4800  5021  5034  5036  5038  5062  5066  5075
          5077  5081  5085  5096  5098  5107  5116  5336  5349  5351  5353  5371  5378  5380
          5391  5393  5402  5411
.RAND      198   200   202   204   209   211   213   215   220   222   224   226   231   233
           235   237   242   244   246   248   253   255   257   259   264   266   268   270
           829   833  1577  1593  1609  1625  1641  1657  1673  1689  1705  1721  1737  1753
          1769  1785  1801  1817  1912  1919  2245  2318  2357  2475  2477  2481  2485  2489
          2493  2497  2501  2505  2509  2513  2517  2521  2525  2529  2533  2537  2541  2550
          2556  2560  2564  2568  2572  2576  2580  2584  2593  2599  2601  2603  2605  2607
          2609  2611  2613  2615  2709  2711  2713  2784  2835  2861  2955  2957  3008  3010
          3071  3073  3083  3087  3091  3095  3101  3107  3111  3115  3119  3123  3129  3135
          3141  3147  3243  3325  3327  3333  3337  3339  3343  3345  3349  3353  3357  3359
          3363  3365  3369  3373  3377  3384  3484  3486  3488  3490  3492  3494  3496  3498
          3500  3502  3504  3506  3508  3510  3512  3514  3724  3726  3734  3747  3759  3766
          3819  4055  4057  4065  4078  4090  4097  4150  4385  4393  4402  4419  4468  4717
          4721  4730  4747  4776  5009  5017  5026  5043  5092  5328  5332  5341  5358  5387
.RB       1577  1579  1584  1586  1593  1595  1600  1602  1609  1611  1616  1618  1625  1627
          1632  1634  1641  1643  1648  1650  1657  1659  1664  1666  1673  1675  1680  1682
          1689  1691  1696  1698  1705  1707  1712  1714  1721  1723  1728  1730  1737  1739
          1744  1746  1753  1755  1760  1762  1769  1771  1776  1778  1785  1787  1792  1794
          1801  1803  1808  1810  1817  1819  1824  1826  2051  2053  2055  2057  2059  2061
          2063  2065  2067  2069  2071  2073  2075  2077  2079  2081  2083  2475  2481  2485
          2489  2493  2497  2501  2505  2509  2513  2517  2521  2525  2550  2556  2558  2560
          2562  2564  2566  2568  2570  2572  2574  2576  2578  2580  2582  2584  2586  2711
          2715  2717  2719  2721  2723  2725  2727  2729  2731  2733  2735  2737  2739  2741
          2743  2745  2747  2749  2751  2753  2755  2757  2759  2761  2763  2765  2767  2769
          2771  2773  2775  2777  2790  2792  2794  2796  2798  2800  2802  2804  2806  2808
          2810  2812  2814  2816  2818  2820  2822  2824  2826  2828  2839  2847  2849  2853
          2855  2857  2859  2957  2963  2965  2967  2969  2971  2973  2975  2977  2979  2981
          2983  2985  2987  2989  2991  2993  2995  2997  2999  3001  3038  3040  3042  3044
          3073  3077  3079  3081  3083  3085  3087  3089  3091  3093  3095  3097  3099  3101
          3103  3105  3107  3109  3111  3113  3115  3117  3119  3121  3123  3125  3127  3129
          3131  3133  3135  3137  3139  3141  3143  3145  3147  3245  3270  3274  3278  3282
          3296  3298  3302  3304  3308  3310  3314  3316  3329  3331  3333  3335  3337  3339
          3341  3343  3345  3347  3349  3351  3353  3355  3357  3359  3361  3363  3365  3367
          3369  3371  3373  3375  3377  3388  3390  3392  3394  3396  3398  3400  3402  3404
          3406  3408  3410  3412  3414  3416  3420  3422  3426  3428  3432  3434  3438  3440
          3444  3446  3452  3454  3456  3458  3460  3462  3464  3466  3468  3470  3472  3474
          3476  3478  3480  3482  3726  3728  3730  3732  3736  3738  3740  3749  3751  3755
          3757  3759  3777  3779  3781  3783  3785  3787  3796  3798  3800  3802  3804  3506
          3808  3817  3823  3830  3832  3839  3841  4057  4059  4061  4063  4067  4069  4071
          4080  4082  4086  4088  4090  4108  4110  4112  4114  4116  4118  4127  4129  4131
          4133  4135  4137  4139  4148  4154  4161  4163  4170  4172  4387  4389  4391  4395
          4397  4404  4406  4410  4412  4414  4430  4432  4436  4438  4440  4442  4447  4451
          4453  4455  4457  4459  4461  4468  4470  4472  4479  4481  4490  4719  4723  4725
          4732  4734  4738  4740  4742  4758  4765  4767  4776  4778  4780  4787  4789  4798
```

```
        5011 5013 5015 5019 5021 5028 5030 5034 5036 5049 5054 5056 5060 5062
        5064 5066 5071 5075 5077 5079 5081 5083 5085 5092 5094 5096 5103 5105
        5114 5330 5334 5336 5343 5345 5349 5351 5353 5369 5376 5378 5387 5389
        5391 5398 5400 5409
.RCONT  1579 1586 1595 1602 1611 1618 1627 1634 1643 1650 1659 1666 1675 1682
        1691 1698 1707 1714 1723 1730 1739 1746 1755 1762 1771 1778 1787 1794
        1803 1810 1819 1826 1914 1921 1923 2053 2055 2057 2059 2061 2063 2065
        2067 2069 2071 2073 2075 2077 2079 2081 2083 2247 2249 2251 2253 2255
        2257 2259 2261 2263 2265 2267 2269 2271 2273 2275 2277 2279 2281 2283
        2285 2287 2289 2291 2293 2295 2297 2299 2301 2303 2305 2307 2309 2320
        2322 2324 2326 2328 2330 2332 2334 2336 2338 2340 2342 2344 2346 2348
        2359 2361 2363 2365 2367 2369 2371 2373 2375 2377 2477 2479 2481 2483
        2485 2487 2489 2491 2493 2495 2497 2499 2501 2503 2505 2507 2509 2511
        2513 2515 2517 2519 2521 2523 2525 2527 2529 2531 2533 2535 2537 2539
        2541 2543 2552 2554 2556 2558 2560 2562 2564 2566 2568 2570 2572 2574
        2576 2578 2580 2582 2584 2586 2595 2597 2599 2601 2603 2605 2607 2609
        2611 2613 2711 2713 2715 2717 2719 2721 2723 2725 2727 2729 2731 2733
        2735 2737 2739 2741 2743 2745 2747 2749 2751 2753 2755 2757 2759 2761
        2763 2765 2767 2769 2771 2773 2775 2777 2786 2788 2790 2792 2794 2796
        2798 2800 2802 2804 2806 2808 2810 2812 2814 2816 2818 2820 2822 2824
        2826 2828 2837 2839 2841 2843 2845 2847 2849 2851 2853 2855 2857 2859
        2957 2959 2961 2963 2965 2967 2969 2971 2973 2975 2977 2979 2981 2983
        2985 2987 2989 2991 2993 2995 2997 2999 3001 3010 3012 3014 3016 3018
        3020 3022 3024 3026 3028 3030 3032 3034 3036 3038 3040 3042 3044 3046
        3048 3050 3052 3054 3056 3058 3060 3062 3064 3073 3075 3077 3079 3081
        3083 3085 3087 3089 3091 3093 3095 3097 3099 3101 3103 3105 3107 3109
        3111 3113 3115 3117 3119 3121 3123 3125 3127 3129 3131 3133 3135 3137
        3139 3141 3143 3145 3147 3245 3247 3250 3252 3254 3256 3258 3260 3262
        3264 3266 3268 3270 3272 3274 3276 3278 3280 3282 3284 3286 3288 3290
        3292 3294 3296 3298 3300 3302 3304 3306 3308 3310 3312 3314 3316 3318
        3327 3329 3331 3333 3335 3337 3339 3341 3343 3345 3347 3349 3351 3353
        3355 3357 3359 3361 3363 3365 3367 3369 3371 3373 3375 3377 3386 3388
        3390 3392 3394 3396 3398 3400 3402 3404 3406 3408 3410 3412 3414 3416
        3418 3420 3422 3424 3426 3428 3430 3432 3434 3436 3438 3440 3442 3444
        3446 3448 3450 3452 3454 3456 3458 3460 3462 3464 3466 3468 3470 3472
        3474 3476 3478 3480 3482 3484 3486 3488 3490 3492 3494 3496 3498 3500
        3502 3504 3506 3508 3510 3512 3514 3726 3728 3730 3732 3734 3736 3738
        3740 3749 3751 3753 3755 3757 3759 3772 3777 3779 3781 3783 3785 3787
        3789 3798 3800 3802 3804 3806 3808 3810 3819 3821 3823 3832 3841 4057
        4059 4061 4063 4065 4067 4069 4071 4080 4082 4084 4086 4088 4090 4103
        4108 4110 4112 4114 4116 4118 4120 4129 4131 4133 4135 4137 4139 4141
        4150 4152 4154 4163 4172 4387 4389 4391 4393 4395 4404 4406 4408 4410
        4412 4425 4430 4432 4434 4436 4438 4440 4449 4451 4453 4455 4457 4459
        4468 4470 4472 4481 4490 4719 4721 4723 4732 4734 4736 4738 4740 4753
        4758 4767 4776 4778 4780 4789 4798 5011 5013 5015 5017 5019 5028 5030
        5032 5034 5036 5049 5054 5056 5058 5060 5062 5064 5073 5075 5077 5079
        5081 5083 5092 5094 5096 5105 5114 5330 5332 5334 5343 5345 5347 5349
        5351 5364 5369 5378 5387 5389 5391 5400 5409
.RCRY   1315 1319 2263 2265 2267 2269 2271 2273 2275 2277 2291 2303 2330 2334
        2336 2340 2342 2363 2367 2373 2489 2493 2497 2501 2517 2521 2533 2541
        2556 2560 2564 2568 2580 2584 2601 2603 2605 2607 2715 2719 2723 2727
        2739 2747 2759 2767 2802 2808 2826 2843 2845 2847 2849 2967 2971 2975
        2981 3014 3016 3018 3022 3032 3042 3050 3052 3062 3081 3085 3089 3093
```

```
            3105    3117    3121    3133    3139    3250    3252    3254    3256    3258    3288    3290    3316    3335      SEQ 1080
            3337    3339    3341    3343    3345    3347    3768    3821    4059    4061    4067    4071    4086    4088
            4099    4108    4110    4112    4114    4116    4118    4120    4129    4131    4133    4135    4137    4139
            4141    4152    4154    4163    4172    4421    4470    4719    4723    4738    4740    4749    4758    4767
            4778    4780    4789    4798    5045    5094    5330    5334    5349    5351    5360    5369    5378    5389
            5391    5400    5409
.RD          198     200     202     204     209     211     213     215     220     222     224     226     231     233
             235     237     242     244     246     248     253     255     257     259     264     266     268     270
             275     277     279     281     283     285     287     391     393     395     397     402     404     406
             408     413     415     417     419     424     426     428     430     435     437     439     441     446
             448     450     452     457     459     461     463     468     470     472     474     476     478     480
             482     484     486     488     490     492     494     499     501     503     505     510     512     514
             516     521     523     525     527     532     534     536     538     543     545     547     549     554
             556     558     560     565     567     569     571     576     578     580     582     584     586     588
             590     592     594     596     598     600     602     607     609     611     613     615     617     619
             621     623     625     627     629     631     633     739     741     743     745     829     831     833
             835     919     921     923     925    1018    1020    1022    1024    1117    1119    1121    1123    1216
            1218    1220    1222    1315    1317    1319    1321    1416    1418    1420    1422    1577    1579    1581
            1584    1586    1588    1593    1595    1597    1600    1602    1604    1609    1611    1613    1616    1618
            1620    1625    1627    1629    1632    1634    1636    1641    1643    1645    1648    1650    1652    1657
            1659    1661    1664    1666    1668    1673    1675    1677    1680    1682    1684    1689    1691    1693
            1696    1698    1700    1705    1707    1709    1712    1714    1716    1721    1723    1725    1728    1730
            1732    1737    1739    1741    1744    1746    1748    1753    1755    1757    1760    1762    1764    1769
            1771    1773    1776    1778    1780    1785    1787    1789    1792    1794    1796    1801    1803    1805
            1808    1810    1812    1817    1819    1821    1824    1826    1828    1912    1914    1916    1919    1921
            1923    1925    2051    2053    2055    2057    2059    2061    2063    2065    2067    2069    2071    2073
            2075    2077    2079    2081    2083    2088    2090    2092    2094    2096    2098    2100    2102    2104
            2106    2108    2110    2112    2114    2116    2118    2120    2245    2247    2249    2251    2253    2255
            2257    2259    2261    2263    2265    2267    2269    2271    2273    2275    2277    2279    2281    2283
            2285    2287    2289    2291    2293    2295    2297    2299    2301    2303    2305    2307    2309    2311
            2313    2318    2320    2322    2324    2326    2328    2330    2332    2334    2336    2338    2340    2342
            2344    2346    2348    2350    2352    2357    2359    2361    2363    2365    2367    2369    2371    2373
            2375    2377    2379    2381    2475    2477    2479    2481    2483    2485    2487    2489    2491    2493
            2495    2497    2499    2501    2503    2505    2507    2509    2511    2513    2515    2517    2519    2521
            2523    2525    2527    2529    2531    2533    2535    2537    2539    2541    2543    2545    2550    2552
            2554    2556    2558    2560    2562    2564    2566    2568    2570    2572    2574    2576    2578    2580
            2582    2584    2586    2588    2593    2595    2597    2599    2601    2603    2605    2607    2609    2611
            2613    2615    2709    2711    2713    2715    2717    2719    2721    2723    2725    2727    2729    2731
            2733    2735    2737    2739    2741    2743    2745    2747    2749    2751    2753    2755    2757    2759
            2761    2763    2765    2767    2769    2771    2773    2775    2777    2779    2784    2786    2788    2790
            2792    2794    2796    2798    2800    2802    2804    2806    2808    2810    2812    2814    2816    2818
            2820    2822    2824    2826    2828    2830    2835    2837    2839    2841    2843    2845    2847    2849
            2851    2853    2855    2857    2859    2861    2955    2957    2959    2961    2963    2965    2967    2969
            2971    2973    2975    2977    2979    2981    2983    2985    2987    2989    2991    2993    2995    2997
            2999    3001    3003    3008    3010    3012    3014    3016    3018    3020    3022    3024    3026    3028
            3030    3032    3034    3036    3038    3040    3042    3044    3046    3048    3050    3052    3054    3056
            3058    3060    3062    3064    3066    3071    3073    3075    3077    3079    3081    3083    3085    3087
            3089    3091    3093    3095    3097    3099    3101    3103    3105    3107    3109    3111    3113    3115
            3117    3119    3121    3123    3125    3127    3129    3131    3133    3135    3137    3139    3141    3143
            3145    3147    3149    3243    3245    3247    3250    3252    3254    3256    3258    3260    3262    3264
            3266    3268    3270    3272    3274    3276    3278    3280    3282    3284    3286    3288    3290    3292
            3294    3296    3298    3300    3302    3304    3306    3308    3310    3312    3314    3316    3318    3320
            3325    3327    3329    3331    3333    3335    3337    3339    3341    3343    3345    3347    3349    3351
```

```
                3353       3355       3357       3359       3361       3363       3365       3367       3369       3371       3373       3375       3377       3379
                3384       3386       3388       3390       3392       3394       3396       3398       3400       3402       3404       3406       3408       3410
                3412       3414       3416       3418       3420       3422       3424       3426       3428       3430       3432       3434       3436       3438
                3440       3442       3444       3446       3448       3450       3452       3454       3456       3458       3460       3462       3464       3466
                3468       3470       3472       3474       3476       3478       3480       3482       3484       3486       3488       3490       3492       3494
                3496       3498       3500       3502       3504       3506       3508       3510       3512       3514       3516       3724       3726       3728
                3730       3732       3734       3736       3738       3740       3742       3747       3749       3751       3753       3755       3757       3759
                3761       3766       3768       3770       3772       3777       3779       3781       3783       3785       3787       3789       3791       3796
                3798       3800       3802       3804       3806       3808       3810       3812       3817       3819       3821       3823       3825       3830
                3832       3834       3839       3841       3843       4055       4057       4059       4061       4063       4065       4067       4069       4071
                4073       4078       4080       4082       4084       4086       4088       4090       4092       4097       4099       4101       4103       4108
                4110       4112       4114       4116       4118       4120       4122       4127       4129       4131       4133       4135       4137       4139
                4141       4143       4148       4150       4152       4154       4156       4161       4163       4165       4170       4172       4174       4385
                4387       4389       4391       4393       4395       4397       4402       4404       4406       4408       4410       4412       4414       4419
                4421       4423       4425       4430       4432       4434       4436       4438       4440       4442       4447       4449       4451       4453
                4455       4457       4459       4461       4466       4468       4470       4472       4474       4479       4481       4483       4488       4490
                4492       4717       4719       4721       4723       4725       4730       4732       4734       4736       4738       4740       4742       4747
                4749       4751       4753       4758       4760       4765       4767       4769       4774       4776       4778       4780       4782       4787
                4789       4791       4796       4798       4800       5009       5011       5013       5015       5017       5019       5021       5026       5028
                5030       5032       5034       5036       5038       5043       5045       5047       5049       5054       5056       5058       5060       5062
                5064       5066       5071       5073       5075       5077       5079       5081       5083       5085       5090       5092       5094       5096
                5098       5103       5105       5107       5112       5114       5116       5328       5330       5332       5334       5336       5341       5343
                5345       5347       5349       5351       5353       5358       5360       5362       5364       5369       5371       5376       5378       5380
                5385       5387       5389       5391       5393       5398       5400       5402       5407       5409       5411
.RJ              198        200        202        204        209        211        213        215        220        222        224        226        231        233
                 235        237        242        244        246        248        253        255        257        259        264        266        268        270
                 275        277        279        281        283        285        287        391        393        395        397        402        404        406
                 408        413        415        417        419        424        426        428        430        435        437        439        441        446
                 448        450        452        457        459        461        463        468        470        472        474        476        478        480
                 482        484        486        488        490        492        494        499        501        503        505 [illegible]        512        514
                 516        521        523        525        527        532        534        536        538        543        545 [illegible]        549        554
                 556        558        560        565        567        569        571        576        578        580        582        584        586        588
                 590        592        594        596        598        600        602        607        609        611        613        615        617        619
                 621        623        625        627        629        631        633        739        741        743        745        829        831        833
                 835        919        921        923        925       1018       1020       1022       1024       1117       1119       1121       1123       1216
                1218       1220       1222       1315       1317       1319       1321       1416       1418       1420       1422       1577       1581       1584
                1588       1593       1597       1600       1604       1609       1613       1616       1620       1625       1629       1632       1636       1641
                1645       1648       1652       1657       1661       1664       1668       1673       1677       1680       1684       1689       1693       1696
                1700       1705       1709       1712       1716       1721       1725       1728       1732       1737       1741       1744       1748       1753
                1757       1760       1764       1769       1773       1776       1780       1785       1789       1792       1796       1801       1805       1808
                1812       1817       1821       1824       1828       1912       1916       1919       1925       2051       2088       2090       2092       2094
                2096       2098       2100       2102       2104       2106       2108       2110       2112       2114       2116       2118       2120       2245
                2311       2313       2318       2350       2352       2357       2379       2381       2475       2545       2550       2588       2593       2615
                2709       2779       2784       2830       2835       2861       2955       3003       3008       3066       3071       3149       3243       3320
                3325       3379       3384       3516       3724       3742       3747       3761       3766       3768       3770       3791       3796       3812
                3817       3825       3830       3834       3839       3843       4055       4073       4078       4092       4097       4099       4101       4122
                4127       4143       4148       4156       4161       4165       4170       4174       4385       4397       4402       4414       4419       4421
                4423       4442       4447       4461       4466       4474       4479       4483       4488       4492       4717       4725       4730       4742
                4747       4749       4751       4760       4765       4769       4774       4782       4787       4791       4796       4800       5009       5021
                5026       5038       5043       5045       5047       5066       5071       5085       5090       5098       5103       5107       5112       5116
                5328       5336       5341       5353       5358       5360       5362       5371       5376       5380       5385       5393       5398       5402
                5407       5411
.RJMAP           198        200        202        204        209        211        213        215        220        222        224        226        231        233
```

```
            235    237    242    244    246    248    253    255    257    259    264    266    268    270
            275    277    279    281    283    285    287    391    393    395    397    402    404    406
            408    413    415    417    419    424    426    428    430    435    437    439    441    446
            448    450    452    457    459    461    463    468    470    472    474    476    478    480
            482    484    486    488    490    492    494    499    501    503    505    510    512    514
            516    521    523    525    527    532    534    536    538    543    545    547    549    554
            556    558    560    565    567    569    571    576    578    580    582    584    586    588
            590    592    594    596    598    600    602    607    609    611    613    615    617    619
            621    623    625    627    629    631    633    739    741    743    745    829    831    833
            835    919    921    923    925   1018   1020   1022   1024   1117   1119   1121   1123   1216
           1218   1220   1222   1315   1317   1319   1321   1416   1418   1420   1422   1577   1581   1584
           1588   1593   1597   1600   1604   1609   1613   1616   1620   1625   1629   1632   1636   1641
           1645   1648   1652   1657   1661   1664   1668   1673   1677   1680   1684   1689   1693   1696
           1700   1705   1709   1712   1716   1721   1725   1728   1732   1737   1741   1744   1748   1753
           1757   1760   1764   1769   1773   1776   1780   1785   1789   1792   1796   1801   1805   1808
           1812   1817   1821   1824   1828   1912   1916   1919   1925   2051   2088   2090   2092   2094
           2096   2098   2100   2102   2104   2106   2108   2110   2112   2114   2116   2118   2120   2245
           2311   2313   2318   2350   2352   2357   2379   2381   2475   2545   2550   2588   2593   2615
           2709   2779   2784   2830   2835   2861   2955   3003   3008   3066   3071   3149   3243   3320
           3325   3379   3384   3516   3724   3742   3747   3761   3766   3768   3770   3791   3796   3812
           3817   3825   3830   3834   3839   3843   4055   4073   4078   4092   4097   4099   4101   4122
           4127   4143   4148   4156   4161   4165   4170   4174   4385   4397   4402   4414   4419   4421
           4423   4442   4447   4461   4466   4474   4479   4483   4488   4492   4717   4725   4730   4742
           4747   4749   4751   4760   4765   4769   4774   4782   4787   4791   4796   4800   5009   5021
           5026   5038   5043   5045   5047   5066   5071   5085   5090   5098   5103   5107   5112   5116
           5328   5336   5341   5353   5358   5360   5362   5371   5376   5380   5385   5393   5398   5402
           5407   5411
.RMGC       198    200    202    204    209    211    213    215    220    222    224    226    231    233
            235    237    242    244    246    248    253    255    257    259    264    266    268    270
            275    277    279    281    283    285    287    393    397    404    408    415    419    426
            430    437    441    448    452    459    463    470    474    478    482    486    490    494
            501    505    512    516    523    527    534    538    545    549    556    560    567    571
            578    582    586    590    594    598    602    609    613    617    621    625    629    633
            739    741    743    745    829    831    833    835    919    921    923    925   1018   1020
           1022   1024   1117   1119   1121   1123   1216   1218   1220   1222   1315   1317   1319   1321
           1416   1418   1420   1422   1579   1581   1586   1588   1595   1597   1602 [illegible] 1611 1613
           1618   1620   1627   1629   1634   1636   1643   1645   1650   1652   1659   1661   1666   1668
           1675   1677   1682   1684   1691   1693   1698   1700   1707   1709   1714   1716   1723   1725
           1730   1732   1739   1741   1746   1748   1755   1757   1762   1764   1771   1773   1778   1780
           1787   1789   1794   1796   1803   1805   1810   1812   1819   1821   1826   1828   1914   1916
           1923   1925   2051   2053   2055   2057   2059   2061   2063   2065   2067   2069   2071   2073
           2075   2077   2079   2081   2083   2088   2090   2092   2094   2096   2098   2100   2102   2104
           2106   2108   2110   2112   2114   2116   2118   2120   2311   2313   2350   2352   2379   2381
           2545   2588   2615   2779   2830   2861   3003   3066   3149   3320   3379   3516   3742   3761
           3791   3812   3825   3834   3843   4073   4092   4122   4143   4156   4165   4174   4397   4414
           4442   4461   4474   4483   4492   4725   4742   4760   4769   4782   4791   4800   5021   5038
           5066   5085   5098   5107   5116   5336   5353   5371   5380   5393   5402   5411
.RNAND      275    277    279    281    283    285    287    919    923   2715   2719   2723   2727   2731
           2735   2739   2743   2747   2751   2755   2759   2763   2767   2771   2775   2790   2794   2798
           2802   2808   2814   2820   2826   2841   2843   2845   2847   2849   2851   2853   2855   2857
           2859
.ROENA      198    200    202    204    209    211    213    215    220    222    224    226    231    233
            235    237    242    244    246    248    253    255    257    259    264    266    268    270
            275    277    279    281    283    285    287    393    397    404    408    415    419    426        SEQ 1083
            430    437    441    448    452    459    463    470    474    478    482    486    490    494
            501    505    512    516    523    527    534    538    545    549    556    560    567    571
            578    582    586    590    594    598    602    609    613    617    621    625    629    633
            741    745    831    835    921    925   1020   1024   1119   1123   1218   1222   1317   1321
           1418   1422   1581   1588   1597   1604   1613   1620   1629   1636   1645   1652   1661   1668
           1677   1684   1693   1700   1709   1716   1725   1732   1741   1748   1757   1764   1773   1780
           1789   1796   1805   1812   1821   1828   1916   1925   2088   2090   2092   2094   2096   2098
           2100   2102   2104   2106   2108   2110   2112   2114   2116   2118   2120   2311   2313   2350
           2352   2379   2381   2545   2588   2615   2779   2830   2861   3003   3066   3149   3320   3379
           3516   3742   3761   3791   3812   3825   3834   3843   4073   4092   4122   4143   4156   4165
           4174   4397   4414   4442   4461   4474   4483   4492   4725   4742   4760   4769   4782   4791
           4800   5021   5038   5066   5085   5098   5107   5116   5336   5353   5371   5380   5393   5402
           5411
.ROR        393    397    404    408    415    419    426    430    437    441    448    452    459    463
            470    474    478    482    486    490    494    501    505    512    516    523    527    534
            538    545    549    556    560    567    571    578    582    586    590    594    598    602
            609    613    617    621    625    629    633    739    741    743    745    831    835    921
            925   1020   1024   1119   1123   1218   1222   1317   1321   1418   1422   1579   1581   1586
           1588   1595   1597   1602   1604   1611   1613   1618   1620   1627   1629   1634   1636   1643
           1645   1650   1652   1659   1661   1666   1668   1675   1677   1682   1684   1691   1693   1698
           1700   1707   1709   1714   1716   1723   1725   1730   1732   1739   1741   1746   1748   1755
           1757   1762   1764   1771   1773   1778   1780   1787   1789   1794   1796   1803   1805   1810
           1812   1819   1821   1826   1828   1914   1916   1923   1925   2051   2053   2055   2057   2059
           2061   2063   2065   2067   2069   2071   2073   2075   2077   2079   2081   2083   2088   2090
           2092   2094   2096   2098   2100   2102   2104   2106   2108   2110   2112   2114   2116   2118
           2120   2247   2249   2251   2253   2255   2257   2259   2261   2263   2265   2267   2269   2271
           2273   2275   2277   2279   2281   2283   2285   2287   2289   2291   2293   2295   2297   2299
           2301   2303   2305   2307   2309   2311   2313   2322   2324   2326   2328   2330   2332   2334
           2336   2338   2340   2342   2344   2346   2348   2350   2352   2361   2363   2365   2367   2369
           2371   2373   2375   2377   2379   2381   2479   2483   2487   2491   2495   2499   2503   2507
           2511   2515   2519   2523   2527   2531   2535   2539   2543   2545   2558   2562   2566   2570
           2574   2578   2582   2586   2588   2717   2721   2725   2729   2733   2737   2741   2745   2749
           2753   2757   2761   2765   2769   2773   2777   2779   2792   2796   2800   2804   2806   2810
           2812   2816   2818   2822   2824   2828   2830   2965   2969   2973   2977   2979   2983   2985
           2989   2991   2995   2997   3001   3003   3066   3097   3103   3125   3131   3137   3143   3149
           3272   3276   3280   3284   3296   3300   3302   3306   3308   3312   3314   3318   3320   3379
           3388   3390   3392   3394   3396   3398   3400   3402   3404   3406   3408   3410   3412   3414
           3416   3418   3516   3732   3738   3742   3761   3770   3772   3791   3812   3825   3830   3834
           3843   4073   4092   4101   4103   4122   4143   4156   4161   4165   4174   4397   4414   4423
           4425   4432   4434   4438   4442   4453   4457   4461   4474   4479   4483   4492   4751   4753
           4760   4769   4782   4787   4791   4800   5021   5038   5047   5049   5056   5058   5062   5066
           5077   5081   5085   5098   5103   5107   5116   5362   5364   5371   5380   5393   5398   5402
           5411
.RPLUS     1216   1220   3728   3730   3736   3740   3755   3757   3768   3777   3779   3781   3783   3785
           3787   3789   3798   3800   3802   3804   3806   3808   3810   3821   3823   3832   3841   4059
           4061   4063   4067   4069   4071   4086   4088   4099   4108   4110   4112   4114   4116   4118
           4120   4129   4131   4133   4135   4137   4139   4141   4152   4154   4163   4172   4421   4470
           4725   4742   4749   4778   5045   5094   5336   5353   5360   5389
.RRMIN      499    503    510    514    521    525    532    536    543    547    554    558    565    569
            576    580    584    588    592    596    600    607    611    615    619    623    627    631
           1315   1319   4387   4389   4391   4395   4410   4412   4430   4436   4440   4451   4455   4459
           4472   4481   4490   4719   4723   4738   4740   4758   4767   4780   4789   4798
```

```
.RSOA     204   215   226   237   248   259   270   468   472   476   480   484   488   492      SEQ 1084
          576   580   584   588   592   596   600  1581  1584  1588  1597  1600  1604  1613
         1616  1620  1629  1632  1636  1645  1648  1652  1661  1664  1668  1677  1680  1684
         1693  1696  1700  1709  1712  1716  1725  1728  1732  1741  1744  1748  1757  1760
         1764  1773  1776  1780  1789  1792  1796  1805  1808  1812  1821  1824  1828  1919
         1921  2088  2090  2092  2094  2096  2098  2100  2102  2104  2106  2108  2110  2112
         2114  2116  2118  2247  2249  2251  2253  2255  2257  2259  2261  2311  2320  2334
         2336  2338  2340  2350  2359  2379  2477  2481  2485  2489  2493  2552  2554  2572
         2576  2580  2584  2588  2595  2597  2763  2767  2771  2775  2779  2786  2788  2806
         2812  2818  2824  2830  2837  2839  2857  2859  2959  2961  2979  2985  2991  2997
         3003  3046  3048  3050  3052  3054  3075  3077  3079  3089  3093  3097  3103  3125
         3131  3137  3143  3245  3247  3286  3288  3290  3292  3294  3296  3302  3308  3314
         3320  3329  3371  3375  3379  3388  3390  3392  3394  3396  3398  3400  3402  3404
         3406  3408  3410  3412  3414  3416  3418  3516  3724  3726  3730  3734  3742  3747
         3749  3751  3753  3761  3766  3768  3770  3772  3777  3779  3781  3796  3798  3800
         3802  3817  3819  3825  3830  3834  3839  3843  4055  4057  4061  4065  4073  4078
         4080  4082  4084  4092  4097  4099  4101  4103  4108  4110  4112  4127  4129  4131
         4133  4148  4150  4156  4161  4165  4170  4174  4385  4387  4393  4402  4404  4406
         4408  4419  4421  4423  4425  4430  4432  4434  4447  4449  4451  4453  4466  4474
         4479  4483  4488  4492  4717  4719  4721  4730  4732  4734  4736  4747  4749  4751
         4753  4758  4760  4765  4767  4769  4774  4782  4787  4791  4796  4800  5009  5011
         5017  5026  5028  5030  5032  5043  5045  5047  5049  5054  5056  5058  5071  5073
         5075  5077  5090  5098  5103  5107  5112  5116  5328  5330  5332  5341  5343  5345
         5347  5358  5360  5362  5364  5369  5371  5376  5378  5380  5385  5393  5398  5402
         5407  5411
.RSOB     202   213   224   235   246   257   268   607   611   615   619   623   627   631
         2295  2297  2299  2301  2303  2305  2307  2309  2342  2344  2346  2348  2529  2533
         2537  2541  2853  2855  3056  3058  3060  3062  3064  3099  3105  3298  3304  3310
         3316  3339  3345  3359  3361  3365  3367  3740  3783  3785  3787  3804  3806  3808
         4071  4114  4116  4118  4135  4137  4139  4389  4436  4455  4468  4470  4776  4778
         5013  5060  5079  5092  5094  5387  5389
.RSOQ     200   211   222   233   244   255   266   391   395   402   406   413   417   424
          428   435   439   446   450   457   461   499   503   510   514   521   525   532
          536   543   547   554   558   565   569   741   745   831   835   921   925  1020
         1024  1119  1123  1218  1222  1317  1321  1418  1422  1914  1916  1923  1925  2120
         2279  2281  2283  2285  2287  2289  2291  2293  2313  2322  2324  2326  2328  2330
         2332  2352  2361  2381  2513  2517  2521  2525  2545  2556  2560  2564  2568  2747
         2751  2755  2759  2851  3038  3040  3042  3044  3081  3085  3149  3270  3274  3278
         3282  3351  3355  3736  3789  3791  3810  3812  3821  4067  4120  4122  4141  4143
         4152  4395  4440  4459  4723  5019  5064  5083  5334
.RSAB    1579  1586  1595  1602  1611  1618  1627  1634  1643  1650  1659  1666  1675  1682
         1691  1698  1707  1714  1723  1730  1739  1746  1755  1762  1771  1778  1787  1794
         1803  1810  1819  1826  2371  2373  2375  2377  2558  2562  2566  2570  2574  2578
         2582  2586  2607  2609  2611  2613  2717  2721  2725  2729  2731  2733  2735  2737
         2739  2741  2743  2745  2749  2753  2757  2761  2765  2769  2773  2777  2792  2796
         2800  2804  2808  2810  2814  2816  2820  2822  2826  2828  2847  2849  2965  2969
         2973  2977  2981  2983  2987  2989  2993  2995  2999  3001  3028  3030  3032  3034
         3036  3083  3087  3091  3095  3101  3107  3111  3115  3119  3123  3127  3129  3133
         3135  3139  3141  3145  3147  3260  3262  3264  3266  3268  3272  3276  3280  3284
         3300  3306  3312  3318  3333  3337  3341  3343  3347  3349  3353  3357  3363  3369
         3373  3377  3386  3452  3454  3456  3458  3460  3462  3464  3466  3468  3470  3472
         3474  3476  3478  3480  3482  3728  3732  3738  3755  3759  3832  3841  4059  4063
         4069  4086  4090  4163  4172  4391  4397  4410  4414  4433  4442  4457  4461  4481
```

```
           4725   4738   4742   4789   5015   5021   5034   5038   5062   5066   5081   5085   5105   5336    SEQ 1085
           5349   5353   5400
.RSAQ       393    397    404    408    415    419    426    430    437    441    448    452    459    463
            470    474    478    482    486    490    494    501    505    512    516    523    527    534
            538    545    549    556    560    567    571    578    582    586    590    594    598    602
            609    613    617    621    625    629    633   2263   2265   2267   2269   2271   2273   2275
           2277   2363   2365   2367   2369   2479   2483   2487   2491   2495   2497   2499   2501   2503
           2505   2507   2509   2511   2515   2519   2523   2527   2531   2535   2539   2543   2599   2601
           2603   2605   2615   2715   2719   2723   2727   2790   2794   2798   2802   2841   2843   2845
           2861   2963   2967   2971   2975   3012   3014   3016   3018   3020   3022   3024   3026   3066
           3109   3113   3117   3121   3250   3252   3254   3256   3258   3331   3335   3420   3422   3424
           3426   3428   3430   3432   3434   3436   3438   3440   3442   3444   3446   3448   3450   3484
           3486   3488   3490   3492   3494   3496   3498   3500   3502   3504   3506   3508   3510   3512
           3514   3757   3823   4088   4154   4412   4472   4490   4740   4780   4798   5036   5096   5114
           5351   5391   5409
.RSDO       198    209    220    231    242    253    264    275    277    279    281    283    285    287
            739    743    829    833    919    923   1018   1022   1117   1121   1216   1220   1315   1319
           1416   1420   1577   1593   1609   1625   1641   1657   1673   1689   1705   1721   1737   1753
           1769   1785   1801   1817   1912   2051   2053   2055   2057   2059   2061   2063   2065   2067
           2069   2071   2073   2075   2077   2079   2081   2083   2245   2318   2357   2475   2550   2593
           2709   2711   2713   2784   2835   2955   2957   3008   3010   3071   3073   3243   3325   3327
           3384
.RSELE      198    200    202    204    209    211    213    215    220    222    224    226    231    233
            235    237    242    244    246    248    253    255    257    259    264    266    268    270
            275    277    279    281    283    285    287    393    397    404    408    415    419    426
            430    437    441    448    452    459    463    470    474    478    482    486    490    494
            501    505    512    516    523    527    534    538    545    549    556    560    567    571
            578    582    586    590    594    598    602    609    613    617    621    625    629    633
            741    745    831    835    921    925   1020   1024   1119   1123   1218   1222   1317   1321
           1418   1422   1581   1588   1597   1604   1613   1620   1629   1636   1645   1652   1661   1668
           1677   1684   1693   1700   1709   1716   1725   1732   1741   1748   1757   1764   1773   1780
           1789   1796   1805   1812   1821   1828   1916   1925   2088   2090   2092   2094   2096   2098
           2100   2102   2104   2106   2108   2110   2112   2114   2116   2118   2120   2311   2313   2350
           2352   2379   2381   2545   2588   2615   2779   2830   2861   3003   3066   3149   3320   3379
           3516   3742   3761   3791   3812   3825   3834   3843   4073   4092   4122   4143   4156   4165
           4174   4397   4414   4442   4461   4474   4483   4492   4725   4742   4760   4769   4782   4791
           4800   5021   5038   5066   5085   5098   5107   5116   5336   5353   5371   5380   5393   5402
           5411
.RSKCN      739    743    829    833    919    923   1018   1022   1117   1121   1216   1220   1315   1319
           1416   1420   1579   1586   1595   1602   1611   1618   1627   1634   1643   1650   1659   1666
           1675   1682   1691   1698   1707   1714   1723   1730   1739   1746   1755   1762   1771   1778
           1787   1794   1803   1810   1819   1826   1914   1923   2051   2053   2055   2057   2059   2061
           2063   2065   2067   2069   2071   2073   2075   2077   2079   2081   2083
.RSMIN     1416   1420   5011   5013   5015   5019   5034   5036   5054   5060   5064   5075   5079   5083
           5096   5105   5114   5330   5334   5349   5351   5369   5378   5391   5400   5409
.RXNOR      391    395    402    406    413    417    424    428    435    439    446    450    457    461
            468    472    476    480    484    488    492   1117   1121   1584   1600   1616   1632   1648
           1664   1680   1696   1712   1728   1744   1760   1776   1792   1808   1824   1921   2320   2359
           2552   2554   2595   2597   2786   2788   2837   2839   2959   2961   3075   3077   3079   3245
           3247   3250   3252   3254   3256   3258   3260   3262   3264   3266   3268   3270   3274   3278
           3282   3286   3288   3290   3292   3294   3298   3304   3310   3316   3329   3331   3335   3341
           3347   3351   3355   3361   3367   3371   3375   3386   3420   3422   3424   3426   3428   3430
           3432   3434   3436   3438   3440   3442   3444   3446   3448   3450   3452   3454   3456   3458
           3460  3462  3464  3466  3468  3470  3472  3474  3476  3478  3480  3482  3749  3751     SEQ 1086
           3753  3796  3817  3839  4080  4082  4084  4127  4148  4170  4404  4406  4408  4447
           4449  4466  4488  4732  4734  4736  4765  4774  4796  5028  5030  5032  5071  5073
           5090  5112  5343  5345  5347  5376  5385  5407
.RXOR      1018  1022  2963  2967  2971  2975  2981  2987  2993  2999  3012  3014  3016  3018
           3020  3022  3024  3026  3028  3030  3032  3034  3036  3038  3040  3042  3044  3046
           3048  3050  3052  3054  3056  3058  3060  3062  3064  3081  3085  3089  3093  3099
           3105  3109  3113  3117  3121  3127  3133  3139  3145
```

```
ATABLE     117    119    121    123    125    127    129    131    133    135    137    139    141    143      SEQ 1087
           145    147    149    151    153    155    157    159    161    163    165    167    169    171
           173    175    177    179    181    183    185    187    360    361    362    364    365    724
           726    728    816    818    820    906    908    910   1004   1006   1008   1103   1105   1107
          1202   1204   1206   1301   1303   1305   1402   1404   1406   1504   1506   1508   1510   1512
          1514   1516   1518   1520   1522   1524   1526   1528   1530   1532   1534   1536   1538   1540
          1542   1544   1546   1548   1550   1552   1554   1556   1558   1560   1562   1564   1566   1568
          1899   1901   1903   2007   2009   2011   2013   2015   2017   2019   2021   2023   2025   2027
          2029   2031   2033   2035   2037   2039   2041   2224   2226   2228   2230   2232   2234   2236
          2460   2462   2464   2466   2694   2696   2698   2700   2940   2942   2944   2946   3228   3230
          3232   3234   3597   3598   3600   3601   3603   3604   3605   3607   3608   3923   3924   3926
          3927   3929   3930   3931   3933   3934   4254   4255   4257   4258   4260   4261   4262   4264
          4265   4572   4573   4575   4576   4578   4579   4580   4582   4583   4880   4881   4883   4884
          4886   4887   4888   4890   4891   5196   5197   5199   5200   5202   5203   5204   5206   5207
CALC       198    200    202    204    209    211    213    215    220    222    224    226    231    233
           235    237    242    244    246    248    253    255    257    259    264    266    268    270
           275    277    279    281    283    285    287    391    393    395    397    402    404    406
           408    413    415    417    419    424    426    428    430    435    437    439    441    446
           448    450    452    457    459    461    463    468    470    472    474    476    478    480
           482    484    486    488    490    492    494    499    501    503    505    510    512    514
           516    521    523    525    527    532    534    536    538    543    545    547    549    554
           556    558    560    565    567    569    571    576    578    580    582    584    586    588
           590    592    594    596    598    600    602    607    609    611    613    615    617    619
           621    623    625    627    629    631    633    739    741    743    745    829    831    833
           835    919    921    923    925   1018   1020   1022   1024   1117   1119   1121   1123   1216
          1218   1220   1222   1315   1317   1319   1321   1416   1418   1420   1422   1577   1579   1581
          1584   1586   1588   1593   1595   1597   1600   1602   1604   1609   1611   1613   1616   1618
          1620   1625   1627   1629   1632   1634   1636   1641   1643   1645   1648   1650   1652   1657
          1659   1661   1664   1666   1668   1673   1675   1677   1680   1682   1684   1689   1691   1693
          1696   1698   1700   1705   1707   1709   1712   1714   1716   1721   1723   1725   1728   1730
          1732   1737   1739   1741   1744   1746   1748   1753   1755   1757   1760   1762   1764   1769
          1771   1773   1776   1778   1780   1785   1787   1789   1792   1794   1796   1801   1803   1805
          1808   1810   1812   1817   1819   1821   1824   1826   1828   1912   1914   1916   1919   1921
          1923   1925   2051   2053   2055   2057   2059   2061   2063   2065   2067   2069   2071   2073
          2075   2077   2079   2081   2083   2088   2090   2092   2094   2096   2098   2100   2102   2104
          2106   2108   2110   2112   2114   2116   2118   2120   2245   2247   2249   2251   2253   2255
          2257   2259   2261   2263   2265   2267   2269   2271   2273   2275   2277   2279   2281   2283
          2285   2287   2289   2291   2293   2295   2297   2299   2301   2303   2305   2307   2309   2311
          2313   2318   2320   2322   2324   2326   2328   2330   2332   2334   2336   2338   2340   2342
          2344   2346   2348   2350   2352   2357   2359   2361   2363   2365   2367   2369   2371   2373
          2375   2377   2379   2381   2475   2477   2479   2481   2483   2485   2487   2489   2491   2493
          2495   2497   2499   2501   2503   2505   2507   2509   2511   2513   2515   2517   2519   2521
          2523   2525   2527   2529   2531   2533   2535   2537   2539   2541   2543   2545   2550   2552
          2554   2556   2558   2560   2562   2564   2566   2568   2570   2572   2574   2576   2578   2580
          2582   2584   2586   2588   2593   2595   2597   2599   2601   2603   2605   2607   2609   2611
          2613   2615   2709   2711   2713   2715   2717   2719   2721   2723   2725   2727   2729   2731
          2733   2735   2737   2739   2741   2743   2745   2747   2749   2751   2753   2755   2757   2759
          2761   2763   2765   2767   2769   2771   2773   2775   2777   2779   2784   2786   2788   2790
          2792   2794   2796   2798   2800   2802   2804   2806   2808   2810   2812   2814   2816   2818
          2820   2822   2824   2826   2828   2830   2835   2837   2839   2841   2843   2845   2847   2849
          2851   2853   2855   2857   2859   2861   2955   2957   2959   2961   2963   2965   2967   2969
          2971   2973   2975   2977   2979   2981   2983   2985   2987   2989   2991   2993   2995   2997
          2999   3001   3003   3008   3010   3012   3014   3016   3018   3020   3022   3024   3026   3028
          3030  3032  3034  3036  3038  3040  3042  3044  3046  3048  3050  3052  3054  3056     SEQ 1088
          3058  3060  3062  3064  3066  3071  3073  3075  3077  3079  3081  3083  3085  3087
          3089  3091  3093  3095  3097  3099  3101  3103  3105  3107  3109  3111  3113  3115
          3117  3119  3121  3123  3125  3127  3129  3131  3133  3135  3137  3139  3141  3143
          3145  3147  3149  3243  3245  3247  3250  3252  3254  3256  3258  3260  3262  3264
          3266  3268  3270  3272  3274  3276  3278  3280  3282  3284  3286  3288  3290  3292
          3294  3296  3298  3300  3302  3304  3306  3308  3310  3312  3314  3316  3318  3320
          3325  3327  3329  3331  3333  3335  3337  3339  3341  3343  3345  3347  3349  3351
          3353  3355  3357  3359  3361  3363  3365  3367  3369  3371  3373  3375  3377  3379
          3384  3386  3388  3390  3392  3394  3396  3398  3400  3402  3404  3406  3408  3410
          3412  3414  3416  3418  3420  3422  3424  3426  3428  3430  3432  3434  3436  3438
          3440  3442  3444  3446  3448  3450  3452  3454  3456  3458  3460  3462  3464  3466
          3468  3470  3472  3474  3476  3478  3480  3482  3484  3486  3488  3490  3492  3494
          3496  3498  3500  3502  3504  3506  3508  3510  3512  3514  3516  3724  3726  3728
          3730  3732  3734  3736  3738  3740  3742  3747  3749  3751  3753  3755  3757  3759
          3761  3766  3768  3770  3772  3777  3779  3781  3783  3785  3787  3789  3791  3796
          3798  3800  3802  3804  3806  3808  3810  3812  3817  3819  3821  3823  3825  3830
          3832  3834  3839  3841  3843  4055  4057  4059  4061  4063  4065  4067  4069  4071
          4073  4078  4080  4082  4084  4086  4088  4090  4092  4097  4099  4101  4103  4108
          4110  4112  4114  4116  4118  4120  4122  4127  4129  4131  4133  4135  4137  4139
          4141  4143  4148  4150  4152  4154  4156  4161  4163  4165  4170  4172  4174  4385
          4387  4389  4391  4393  4395  4397  4402  4404  4406  4408  4410  4412  4414  4419
          4421  4423  4425  4430  4432  4434  4436  4438  4440  4442  4447  4449  4451  4453
          4455  4457  4459  4461  4466  4468  4470  4472  4474  4479  4481  4483  4488  4490
          4492  4717  4719  4721  4723  4725  4730  4732  4734  4736  4738  4740  4742  4747
          4749  4751  4753  4758  4760  4765  4767  4769  4774  4776  4778  4780  4782  4787
          4789  4791  4796  4798  4800  5009  5011  5013  5015  5017  5019  5021  5026  5028
          5030  5032  5034  5036  5038  5043  5045  5047  5049  5054  5056  5058  5060  5062
          5064  5066  5071  5073  5075  5077  5079  5081  5083  5085  5090  5092  5094  5096
          5098  5103  5105  5107  5112  5114  5116  5328  5330  5332  5334  5336  5341  5343
          5345  5347  5349  5351  5353  5358  5360  5362  5364  5369  5371  5376  5378  5380
          5385  5387  5389  5391  5393  5398  5400  5402  5407  5409  5411
CONCAT     198   200   202   204   209   211   213   215   220   222   224   226   231   233
           235   237   242   244   246   248   253   255   257   259   264   266   268   270
           275   277   279   281   283   285   287   391   393   395   397   402   404   406
           408   413   415   417   419   424   426   428   430   435   437   439   441   446
           448   450   452   457   459   461   463   468   470   472   474   476   478   480
           482   484   486   488   490   492   494   499   501   503   505   510   512   514
           516   521   523   525   527   532   534   536   538   543   545   547   549   554
           556   558   560   565   567   569   571   576   578   580   582   584   586   588
           590   592   594   596   598   600   602   607   609   611   613   615   617   619
           621   623   625   627   629   631   633   739   741   743   745   829   831   833
           835   919   921   923   925  1018  1020  1022  1024  1117  1119  1121  1123  1216
          1218  1220  1222  1315  1317  1319  1321  1416  1418  1420  1422  1577  1579  1581
          1584  1586  1588  1593  1595  1597  1600  1602  1604  1609  1611  1613  1616  1618
          1620  1625  1627  1629  1632  1634  1636  1641  1643  1645  1648  1650  1652  1657
          1659  1661  1664  1666  1668  1673  1675  1677  1680  1682  1684  1689  1691  1693
          1696  1698  1700  1705  1707  1709  1712  1714  1716  1721  1723  1725  1728  1730
          1732  1737  1739  1741  1744  1746  1748  1753  1755  1757  1760  1762  1764  1769
          1771  1773  1776  1778  1780  1785  1787  1789  1792  1794  1796  1801  1803  1805
          1808  1810  1812  1817  1819  1821  1824  1826  1828  1912  1914  1916  1919  1921
          1923  1925  2051  2053  2055  2057  2059  2061  2063  2065  2067  2069  2071  2073
          2075  2077  2079  2081  2083  2088  2090  2092  2094  2096  2098  2100  2102  2104
```

```
1076     2106  2108  2110  2112  2114  2116  2118  2120  2245  2247  2249  2251  2253  2255     SEQ 1089
         2257  2259  2261  2263  2265  2267  2269  2271  2273  2275  2277  2279  2281  2283
         2285  2287  2289  2291  2293  2295  2297  2299  2301  2303  2305  2307  2309  2311
         2313  2318  2320  2322  2324  2326  2328  2330  2332  2334  2336  2338  2340  2342
         2344  2346  2348  2350  2352  2357  2359  2361  2363  2365  2367  2369  2371  2373
         2375  2377  2379  2381  2475  2477  2479  2481  2483  2485  2487  2489  2491  2493
         2495  2497  2499  2501  2503  2505  2507  2509  2511  2513  2515  2517  2519  2521
         2523  2525  2527  2529  2531  2533  2535  2537  2539  2541  2543  2545  2550  2552
         2554  2556  2558  2560  2562  2564  2566  2568  2570  2572  2574  2576  2578  2580
         2582  2584  2586  2588  2593  2595  2597  2599  2601  2603  2605  2607  2609  2611
         2613  2615  2709  2711  2713  2715  2717  2719  2721  2723  2725  2727  2729  2731
         2733  2735  2737  2739  2741  2743  2745  2747  2749  2751  2753  2755  2757  2759
         2761  2763  2765  2767  2769  2771  2773  2775  2777  2779  2784  2786  2788  2790
         2792  2794  2796  2798  2800  2802  2804  2806  2808  2810  2812  2814  2816  2818
         2820  2822  2824  2826  2828  2830  2835  2837  2839  2841  2843  2845  2847  2849
         2851  2853  2855  2857  2859  2861  2955  2957  2959  2961  2963  2965  2967  2969
         2971  2973  2975  2977  2979  2981  2983  2985  2987  2989  2991  2993  2995  2997
         2999  3001  3003  3008  3010  3012  3014  3016  3018  3020  3022  3024  3026  3028
         3030  3032  3034  3036  3038  3040  3042  3044  3046  3048  3050  3052  3054  3056
         3058  3060  3062  3064  3066  3071  3073  3075  3077  3079  3081  3083  3085  3087
         3089  3091  3093  3095  3097  3099  3101  3103  3105  3107  3109  3111  3113  3115
         3117  3119  3121  3123  3125  3127  3129  3131  3133  3135  3137  3139  3141  3143
         3145  3147  3149  3243  3245  3247  3250  3252  3254  3256  3258  3260  3262  3264
         3266  3268  3270  3272  3274  3276  3278  3280  3282  3284  3286  3288  3290  3292
         3294  3296  3298  3300  3302  3304  3306  3308  3310  3312  3314  3316  3318  3320
         3325  3327  3329  3331  3333  3335  3337  3339  3341  3343  3345  3347  3349  3351
         3353  3355  3357  3359  3361  3363  3365  3367  3369  3371  3373  3375  3377  3379
         3384  3386  3388  3390  3392  3394  3396  3398  3400  3402  3404  3406  3408  3410
         3412  3414  3416  3418  3420  3422  3424  3426  3428  3430  3432  3434  3436  3438
         3440  3442  3444  3446  3448  3450  3452  3454  3456  3458  3460  3462  3464  3466
         3468  3470  3472  3474  3476  3478  3480  3482  3484  3486  3488  3490  3492  3494
         3496  3498  3500  3502  3504  3506  3508  3510  3512  3514  3516  3724  3726  3728
         3730  3732  3734  3736  3738  3740  3742  3747  3749  3751  3753  3755  3757  3759
         3761  3766  3768  3770  3772  3777  3779  3781  3783  3785  3787  3789  3791  3796
         3798  3800  3802  3804  3806  3808  3810  3812  3817  3819  3821  3823  3825  3830
         3832  3834  3839  3841  3843  4055  4057  4059  4061  4063  4065  4067  4069  4071
         4073  4078  4080  4082  4084  4086  4088  4090  4092  4097  4099  4101  4103  4108
         4110  4112  4114  4116  4118  4120  4122  4127  4129  4131  4133  4135  4137  4139
         4141  4143  4148  4150  4152  4154  4156  4161  4163  4165  4170  4172  4174  4385
         4387  4389  4391  4393  4395  4397  4402  4404  4406  4408  4410  4412  4414  4419
         4421  4423  4425  4430  4432  4434  4436  4438  4440  4442  4447  4449  4451  4453
         4455  4457  4459  4461  4466  4468  4470  4472  4474  4479  4481  4483  4488  4490
         4492  4717  4719  4721  4723  4725  4730  4732  4734  4736  4738  4740  4742  4747
         4749  4751  4753  4758  4760  4765  4767  4769  4774  4776  4778  4780  4782  4787
         4789  4791  4796  4798  4800  5009  5011  5013  5015  5017  5019  5021  5026  5028
         5030  5032  5034  5036  5038  5043  5045  5047  5049  5054  5056  5058  5060  5062
         5064  5066  5071  5073  5075  5077  5079  5081  5083  5085  5090  5092  5094  5096
         5098  5103  5105  5107  5112  5114  5116  5328  5330  5332  5334  5336  5341  5343
         5345  5347  5349  5351  5353  5358  5360  5362  5364  5369  5371  5376  5378  5380
         5385  5387  5389  5391  5393  5398  5400  5402  5407  5409  5411
FIELD     198   200   202   204   209   211   213   215   220   222   224   226   231   233
          235   237   242   244   246   248   253   255   257   259   264   266   268   270
          275   277   279   281   283   285   287   391   393   395   397   402   404   406
```

1077

408
448
482
516
556
590
621
835
1218
1584
1620
1659
1696
1732
1771
1808
1923
2075
2106
2257
2285
2313
2344
2375
2495
2523
2554
2582
2613
2733
2761
2792
2820
2851
2971
2999
3030
3058
3089
3117
3145
3266
3294
3325
3353
3384
3412
3440
3468
3496
3730
3761
3798

413
450
484
521
558
592
623
919
1220
1586
1625
1661
1698
1737
1773
1810
1925
2077
2108
2259
2287
2318
2346
2377
2497
2525
2556
2584
2615
2735
2763
2794
2822
2853
2973
3001
3032
3060
3091
3119
3147
3268
3296
3327
3355
3386
3414
3442
3470
3498
3732
3766
3800

415
452
486
523
560
594
625
921
1222
1588
1627
1664
1700
1739
1776
1812
2051
2079
2110
2261
2289
2320
2348
2379
2499
2527
2558
2586
2709
2737
2765
2796
2824
2855
2975
3003
3034
3062
3093
3121
3149
3270
3298
3329
3357
3388
3416
3444
3472
3500
3734
3768
3802

417
457
488
525
565
596
627
923
1315
1593
1629
1666
1705
1741
1778
1817
2053
2081
2112
2263
2291
2322
2350
2381
2501
2529
2560
2588
2711
2739
2767
2798
2826
2857
2977
3008
3036
3064
3095
3123
3243
3272
3300
3331
3359
3390
3418
3446
3474
3502
3736
3770
3804

419
459
490
527
567
598
629
925
1317
1595
1632
1668
1707
1744
1780
1819
2055
2083
2114
2265
2293
2324
2352
2475
2503
2531
2562
2593
2713
2741
2769
2800
2828
2859
2979
3010
3038
3066
3097
3125
3245
3274
3302
3333
3361
3392
3420
3448
3476
3504
3738
3772
3806

424
461
492
532
569
600
631
1018
1319
1597
1634
1673
1709
1746
1785
1821
2057
2088
2116
2267
2295
2326
2357
2477
2505
2533
2564
2595
2715
2743
2771
2802
2830
2861
2981
3012
3040
3071
3099
3127
3247
3276
3304
3335
3363
3394
3422
3450
3478
3506
3740
3777
3808

426
463
494
534
571
602
633
1020
1321
1600
1636
1675
1712
1748
1787
1824
2059
2090
2118
2269
2297
2328
2359
2479
2507
2535
2566
2597
2717
2745
2773
2804
2835
2955
2983
3014
3042
3073
3101
3129
3250
3278
3306
3337
3365
3396
3424
3452
3480
3508
3742
3779
3810

428
468
499
536
576
607
739
1022
1416
1602
1641
1677
1714
1753
1789
1826
2061
2092
2120
2271
2299
2330
2361
2481
2509
2537
2568
2599
2719
2747
2775
2806
2837
2957
2985
3016
3044
3075
3103
3131
3252
3280
3308
3339
3367
3398
3426
3454
3482
3510
3747
3781
3812

430
470
501
538
578
609
741
1024
1418
1604
1643
1680
1716
1755
1792
1828
2063
2094
2245
2273
2301
2332
2363
2483
2511
2539
2570
2601
2721
2749
2777
2808
2839
2959
2987
3018
3046
3077
3105
3133
3254
3282
3310
3341
3369
3400
3428
3456
3484
3512
3749
3783
3817

435
472
503
543
580
611
743
1117
1420
1609
1645
1682
1721
1757
1794
1912
2065
2096
2247
2275
2303
2334
2365
2485
2513
2541
2572
2603
2723
2751
2779
2810
2841
2961
2989
3020
3048
3079
3107
3135
3256
3284
3312
3343
3371
3402
3430
3458
3486
3514
3751
3785
3819

437
474
505
545
582
613
745
1119
1422
1611
1648
1684
1723
1760
1796
1914
2067
2098
2249
2277
2305
2336
2367
2487
2515
2543
2574
2605
2725
2753
2784
2812
2843
2963
2991
3022
3050
3081
3109
3137
3258
3286
3314
3345
3373
3404
3432
3460
3488
3516
3753
3787
3821

439
476
510
547
584
615
829
1121
1577
1613
1650
1689
1725
1762
1801
1916
2069
2100
2251
2279
2307
2338
2369
2489
2517
2545
2576
2607
2727
2755
2786
2814
2845
2965
2993
3024
3052
3083
3111
3139
3260
3288
3316
3347
3375
3406
3434
3462
3490
3724
3755
3789
3823

441
478
512
549
586
617
831
1123
1579
1616
1652
1691
1728
1764
1803
1919
2071
2102
2253
2281
2309
2340
2371
2491
2519
2550
2578
2609
2729
2757
2788
2816
2847
2967
2995
3026
3054
3085
3113
3141
3262
3290
3318
3349
3377
3408
3436
3464
3492
3726
3757
3791
3825

446
480
514
554
588
619
833
1216
1581
1618
1657
1693
1730
1769
1805
1921
2073
2104
2255
2283
2311
2342
2373
2493
2521
2552
2580
2611
2731
2759
2790
2818
2849
2969
2997
3028
3056
3087
3115
3143
3264
3292
3320
3351
3379
3410
3438
3466
3494
3728
3759
3796
3830

SEQ 1090

DFP
DFP

```
         3832    3834    3839    3841    3843    4055    4057    4059    4061    4063    4065    4067    4069    4071
         4073    4078    4080    4082    4084    4086    4088    4090    4092    4097    4099    4101    4103    4108
         4110    4112    4114    4116    4118    4120    4122    4127    4129    4131    4133    4135    4137    4139
         4141    4143    4148    4150    4152    4154    4156    4161    4163    4165    4170    4172    4174    4385
         4387    4389    4391    4393    4395    4397    4402    4404    4406    4408    4410    4412    4414    4419
         4421    4423    4425    4430    4432    4434    4436    4438    4440    4442    4447    4449    4451    4453
         4455    4457    4459    4461    4466    4468    4470    4472    4474    4479    4481    4483    4488    4490
         4492    4717    4719    4721    4723    4725    4730    4732    4734    4736    4738    4740    4742    4747
         4749    4751    4753    4758    4760    4765    4767    4769    4774    4776    4778    4780    4782    4787
         4789    4791    4796    4798    4800    5009    5011    5013    5015    5017    5019    5021    5026    5028
         5030    5032    5034    5036    5038    5043    5045    5047    5049    5054    5056    5058    5060    5062
         5064    5066    5071    5073    5075    5077    5079    5081    5083    5085    5090    5092    5094    5096
         5098    5103    5105    5107    5112    5114    5116    5328    5330    5332    5334    5336    5341    5343
         5345    5347    5349    5351    5353    5358    5360    5362    5364    5369    5371    5376    5378    5380
         5385    5387    5389    5391    5393    5398    5400    5402    5407    5409    5411
GET      5464    5465
GO         85      87      93      99     328     330     336     342     686     688     694     700     784     786
          792     798     874     876     882     888     966     968     974     980    1065    1067    1073    1079
         1164    1166    1172    1178    1263    1265    1271    1277    1364    1366    1372    1378    1472    1474
         1480    1486    1867    1869    1875    1881    1969    1971    1977    1983    2192    2194    2200    2206
         2428    2430    2436    2442    2662    2664    2670    2676    2908    2910    2916    2922    3196    3198
         3204    3210    3565    3567    3573    3579    3684    3687    3691    3702    3707    3891    3893    3899
         3905    4012    4019    4023    4025    4030    4032    4038    4222    4224    4230    4236    4341    4345
         4351    4362    4367    4540    4542    4548    4554    4661    4671    4680    4690    4695    4848    4850
         4856    4862    4967    4970    4972    4976    4978    4984    4986    4991    4996    5164    5166    5172
         5178    5283    5287    5291    5293    5295    5301    5303    5305    5310    5315    5317
MFLD      198#    198     200#    200     202#    202     204#    204     209#    209     211#    211     213#    213
          215#    215     220#    220     222#    222     224#    224     226#    226     231#    231     233#    233
          235#    235     237#    237     242#    242     244#    244     246#    246     248#    248     253#    253
          255#    255     257#    257     259#    259     264#    264     266#    266     268#    268     270#    270
          275#    275     277#    277     279#    279     281#    281     283#    283     285#    285     287#    287
          391#    391     393#    393     395#    395     397#    397     402#    402     404#    404     406#    406
          408#    408     413#    413     415#    415     417#    417     419#    419     424#    424     426#    426
          428#    428     430#    430     435#    435     437#    437     439#    439     441#    441     446#    446
          448#    448     450#    450     452#    452     457#    457     459#    459     461#    461     463#    463
          468#    468     470#    470     472#    472     474#    474     476#    476     478#    478     480#    480
          482#    482     484#    484     486#    486     488#    488     490#    490     492#    492     494#    494
          499#    499     501#    501     503#    503     505#    505     510#    510     512#    512     514#    514
          516#    516     521#    521     523#    523     525#    525     527#    527     532#    532     534#    534
          536#    536     538#    538     543#    543     545#    545     547#    547     549#    549     554#    554
          556#    556     558#    558     560#    560     565#    565     567#    567     569#    569     571#    571
          576#    576     578#    578     580#    580     582#    582     584#    584     586#    586     588#    588
          590#    590     592#    592     594#    594     596#    596     598#    598     600#    600     602#    602
          607#    607     609#    609     611#    611     613#    613     615#    615     617#    617     619#    619
          621#    621     623#    623     625#    625     627#    627     629#    629     631#    631     633#    633
          739#    739     741#    741     743#    743     745#    745     829#    829     831#    831     833#    833
          835#    835     919#    919     921#    921     923#    923     925#    925    1018#   1018    1020#   1020
         1022#   1022    1024#   1024      17#   1117    1119#   1119    1121#   1121    1123#   1123    1216#   1216
         1218#   1218    1220#   1220       2#   1222    1315#   1315    1317#   1317    1319#   1319    1321#   1321
         1416#   1416    1418#   1418      20#   1420    1422#   1422    1577#   1577    1579#   1579    1581#   1581
         1584#   1584    1586#   1586     588#   1588    1593#   1593    1595#   1595    1597#   1597    1600#   1600
         1602#   1602    1604#   1604    1609#   1609    1611#   1611    1613#   1613    1616#   1616    1618#   1618
         1620#   1620    1625#   1625    1627#   1627    1629#   1629    1632#   1632    1634#   1634    1636#   1636
```

```
1641#  1641    1643#  1643    1645#  1645    1648#  1648    1650#  1650    1652#  1652    1657#  1657
1659#  1659    1661#  1661    1664#  1664    1666#  1666    1668#  1668    1673#  1673    1675#  1675
1677#  1677    1680#  1680    1682#  1682    1684#  1684    1689#  1689    1691#  1691    1693#  1693
1696#  1696    1698#  1698    1700#  1700    1705#  1705    1707#  1707    1709#  1709    1712#  1712
1714#  1714    1716#  1716    1721#  1721    1723#  1723    1725#  1725    1728#  1728    1730#  1730
1732#  1732    1737#  1737    1739#  1739    1741#  1741    1744#  1744    1746#  1746    1748#  1748
1753#  1753    1755#  1755    1757#  1757    1760#  1760    1762#  1762    1764#  1764    1769#  1769
1771#  1771    1773#  1773    1776#  1776    1778#  1778    1780#  1780    1785#  1785    1787#  1787
1789#  1789    1792#  1792    1794#  1794    1796#  1796    1801#  1801    1803#  1803    1805#  1805
1808#  1808    1810#  1810    1812#  1812    1817#  1817    1819#  1819    1821#  1821    1824#  1824
1826#  1826    1828#  1828    1912#  1912    1914#  1914    1916#  1916    1919#  1919    1921#  1921
1923#  1923    1925#  1925    2051#  2051    2053#  2053    2055#  2055    2057#  2057    2059#  2059
2061#  2061    2063#  2063    2065#  2065    2067#  2067    2069#  2069    2071#  2071    2073#  2073
2075#  2075    2077#  2077    2079#  2079    2081#  2081    2083#  2083    2088#  2088    2090#  2090
2092#  2092    2094#  2094    2096#  2096    2098#  2098    2100#  2100    2102#  2102    2104#  2104
2106#  2106    2108#  2108    2110#  2110    2112#  2112    2114#  2114    2116#  2116    2118#  2118
2120#  2120    2245#  2245    2247#  2247    2249#  2249    2251#  2251    2253#  2253    2255#  2255
2257#  2257    2259#  2259    2261#  2261    2263#  2263    2265#  2265    2267#  2267    2269#  2269
2271#  2271    2273#  2273    2275#  2275    2277#  2277    2279#  2279    2281#  2281    2283#  2283
2285#  2285    2287#  2287    2289#  2289    2291#  2291    2293#  2293    2295#  2295    2297#  2297
2299#  2299    2301#  2301    2303#  2303    2305#  2305    2307#  2307    2309#  2309    2311#  2311
2313#  2313    2318#  2318    2320#  2320    2322#  2322    2324#  2324    2326#  2326    2328#  2328
2330#  2330    2332#  2332    2334#  2334    2336#  2336    2338#  2338    2340#  2340    2342#  2342
2344#  2344    2346#  2346    2348#  2348    2350#  2350    2352#  2352    2357#  2357    2359#  2359
2361#  2361    2363#  2363    2365#  2365    2367#  2367    2369#  2369    2371#  2371    2373#  2373
2375#  2375    2377#  2377    2379#  2379    2381#  2381    2475#  2475    2477#  2477    2479#  2479
2481#  2481    2483#  2483    2485#  2485    2487#  2487    2489#  2489    2491#  2491    2493#  2493
2495#  2495    2497#  2497    2499#  2499    2501#  2501    2503#  2503    2505#  2505    2507#  2507
2509#  2509    2511#  2511    2513#  2513    2515#  2515    2517#  2517    2519#  2519    2521#  2521
2523#  2523    2525#  2525    2527#  2527    2529#  2529    2531#  2531    2533#  2533    2535#  2535
2537#  2537    2539#  2539    2541#  2541    2543#  2543    2545#  2545    2550#  2550    2552#  2552
2554#  2554    2556#  2556    2558#  2558    2560#  2560    2562#  2562    2564#  2564    2566#  2566
2568#  2568    2570#  2570    2572#  2572    2574#  2574    2576#  2576    2578#  2578    2580#  2580
2582#  2582    2584#  2584    2586#  2586    2588#  2588    2593#  2593    2595#  2595    2597#  2597
2599#  2599    2601#  2601    2603#  2603    2605#  2605    2607#  2607    2609#  2609    2611#  2611
2613#  2613    2615#  2615    2709#  2709    2711#  2711    2713#  2713    2715#  2715    2717#  2717
2719#  2719    2721#  2721    2723#  2723    2725#  2725    2727#  2727    2729#  2729    2731#  2731
2733#  2733    2735#  2735    2737#  2737    2739#  2739    2741#  2741    2743#  2743    2745#  2745
2747#  2747    2749#  2749    2751#  2751    2753#  2753    2755#  2755    2757#  2757    2759#  2759
2761#  2761    2763#  2763    2765#  2765    2767#  2767    2769#  2769    2771#  2771    2773#  2773
2775#  2775    2777#  2777    2779#  2779    2784#  2784    2786#  2786    2788#  2788    2790#  2790
2792#  2792    2794#  2794    2796#  2796    2798#  2798    2800#  2800    2802#  2802    2804#  2804
2806#  2806    2808#  2808    2810#  2810    2812#  2812    2814#  2814    2816#  2816    2818#  2818
2820#  2820    2822#  2822    2824#  2824    2826#  2826    2828#  2828    2830#  2830    2835#  2835
2837#  2837    2839#  2839    2841#  2841    2843#  2843    2845#  2845    2847#  2847    2849#  2849
2851#  2851    2853#  2853    2855#  2855    2857#  2857    2859#  2859    2861#  2861    2955#  2955
2957#  2957    2959#  2959    2961#  2961    2963#  2963    2965#  2965    2967#  2967    2969#  2969
2971#  2971    2973#  2973    2975#  2975    2977#  2977    2979#  2979    2981#  2981    2983#  2983
2985#  2985    2987#  2987    2989#  2989    2991#  2991    2993#  2993    2995#  2995    2997#  2997
2999#  2999    3001#  3001    3003#  3003    3008#  3008    3010#  3010    3012#  3012    3014#  3014
3016#  3016    3018#  3018    3020#  3020    3022#  3022    3024#  3024    3026#  3026    3028#  3028
3030#  3030    3032#  3032    3034#  3034    3036#  3036    3038#  3038    3040#  3040    3042#  3042
3044#  3044    3046#  3046    3048#  3048    3050#  3050    3052#  3052    3054#  3054    3056#  3056
```

```
3058#   3058    3060#   3060    3062#   3062    3064#   3064    3066#   3066    3071#   3071    3073#   3073     SEQ 1093
3075#   3075    3077#   3077    3079#   3079    3081#   3081    3083#   3083    3085#   3085    3087#   3087
3089#   3089    3091#   3091    3093#   3093    3095#   3095    3097#   3097    3099#   3099    3101#   3101
3103#   3103    3105#   3105    3107#   3107    3109#   3109    3111#   3111    3113#   3113    3115#   3115
3117#   3117    3119#   3119    3121#   3121    3123#   3123    3125#   3125    3127#   3127    3129#   3129
3131#   3131    3133#   3133    3135#   3135    3137#   3137    3139#   3139    3141#   3141    3143#   3143
3145#   3145    3147#   3147    3149#   3149    3243#   3243    3245#   3245    3247#   3247    3250#   3250
3252#   3252    3254#   3254    3256#   3256    3258#   3258    3260#   3260    3262#   3262    3264#   3264
3266#   3266    3268#   3268    3270#   3270    3272#   3272    3274#   3274    3276#   3276    3278#   3278
3280#   3280    3282#   3282    3284#   3284    3286#   3286    3288#   3288    3290#   3290    3292#   3292
3294#   3294    3296#   3296    3298#   3298    3300#   3300    3302#   3302    3304#   3304    3306#   3306
3308#   3308    3310#   3310    3312#   3312    3314#   3314    3316#   3316    3318#   3318    3320#   3320
3325#   3325    3327#   3327    3329#   3329    3331#   3331    3333#   3333    3335#   3335    3337#   3337
3339#   3339    3341#   3341    3343#   3343    3345#   3345    3347#   3347    3349#   3349    3351#   3351
3353#   3353    3355#   3355    3357#   3357    3359#   3359    3361#   3361    3363#   3363    3365#   3365
3367#   3367    3369#   3369    3371#   3371    3373#   3373    3375#   3375    3377#   3377    3379#   3379
3384#   3384    3386#   3386    3388#   3388    3390#   3390    3392#   3392    3394#   3394    3396#   3396
3398#   3398    3400#   3400    3402#   3402    3404#   3404    3406#   3406    3408#   3408    3410#   3410
3412#   3412    3414#   3414    3416#   3416    3418#   3418    3420#   3420    3422#   3422    3424#   3424
3426#   3426    3428#   3428    3430#   3430    3432#   3432    3434#   3434    3436#   3436    3438#   3438
3440#   3440    3442#   3442    3444#   3444    3446#   3446    3448#   3448    3450#   3450    3452#   3452
3454#   3454    3456#   3456    3458#   3458    3460#   3460    3462#   3462    3464#   3464    3466#   3466
3468#   3468    3470#   3470    3472#   3472    3474#   3474    3476#   3476    3478#   3478    3480#   3480
3482#   3482    3484#   3484    3486#   3486    3488#   3488    3490#   3490    3492#   3492    3494#   3494
3496#   3496    3498#   3498    3500#   3500    3502#   3502    3504#   3504    3506#   3506    3508#   3508
3510#   3510    3512#   3512    3514#   3514    3516#   3516    3724#   3724    3726#   3726    3728#   3728
3730#   3730    3732#   3732    3734#   3734    3736#   3736    3738#   3738    3740#   3740    3742#   3742
3747#   3747    3749#   3749    3751#   3751    3753#   3753    3755#   3755    3757#   3757    3759#   3759
3761#   3761    3766#   3766    3768#   3768    3770#   3770    3772#   3772    3777#   3777    3779#   3779
3781#   3781    3783#   3783    3785#   3785    3787#   3787    3789#   3789    3791#   3791    3796#   3796
3798#   3798    3800#   3800    3802#   3802    3804#   3804    3806#   3806    3808#   3808    3810#   3810
3812#   3812    3817#   3817    3819#   3819    3821#   3821    3823#   3823    3825#   3825    3830#   3830
3832#   3832    3834#   3834    3839#   3839    3841#   3841    3843#   3843    4055#   4055    4057#   4057
4059#   4059    4061#   4061    4063#   4063    4065#   4065    4067#   4067    4069#   4069    4071#   4071
4073#   4073    4078#   4078    4080#   4080    4082#   4082    4084#   4084    4086#   4086    4088#   4088
4090#   4090    4092#   4092    4097#   4097    4099#   4099    4101#   4101    4103#   4103    4108#   4108
4110#   4110    4112#   4112    4114#   4114    4116#   4116    4118#   4118    4120#   4120    4122#   4122
4127#   4127    4129#   4129    4131#   4131    4133#   4133    4135#   4135    4137#   4137    4139#   4139
4141#   4141    4143#   4143    4148#   4148    4150#   4150    4152#   4152    4154#   4154    4156#   4156
4161#   4161    4163#   4163    4165#   4165    4170#   4170    4172#   4172    4174#   4174    4385#   4385
4387#   4387    4389#   4389    4391#   4391    4393#   4393    4395#   4395    4397#   4397    4402#   4402
4404#   4404    4406#   4406    4408#   4408    4410#   4410    4412#   4412    4414#   4414    4419#   4419
4421#   4421    4423#   4423    4425#   4425    4430#   4430    4432#   4432    4434#   4434    4436#   4436
4438#   4438    4440#   4440    4442#   4442    4447#   4447    4449#   4449    4451#   4451    4453#   4453
4455#   4455    4457#   4457    4459#   4459    4461#   4461    4466#   4466    4468#   4468    4470#   4470
4472#   4472    4474#   4474    4479#   4479    4481#   4481    4483#   4483    4488#   4488    4490#   4490
4492#   4492    4717#   4717    4719#   4719    4721#   4721    4723#   4723    4725#   4725    4730#   4730
4732#   4732    4734#   4734    4736#   4736    4738#   4738    4740#   4740    4742#   4742    4747#   4747
4749#   4749    4751#   4751    4753#   4753    4758#   4758    4760#   4760    4765#   4765    4767#   4767
4769#   4769    4774#   4774    4776#   4776    4778#   4778    4780#   4780    4782#   4782    4787#   4787
4789#   4789    4791#   4791    4796#   4796    4798#   4798    4800#   4800    5009#   5009    5011#   5011
5013#   5013    5015#   5015    5017#   5017    5019#   5019    5021#   5021    5026#   5026    5028#   5028
5030#   5030    5032#   5032    5034#   5034    5036#   5036    5038#   5038    5043#   5043    5045#   5045
```

```
        5047#  5047   5049#  5049   5054#  5054   5056#  5056   5058#  5058   5060#  5060   5062#  5062       SEQ 1094
        5064#  5064   5066#  5066   5071#  5071   5073#  5073   5075#  5075   5077#  5077   5079#  5079
        5081#  5081   5083#  5083   5085#  5085   5090#  5090   5092#  5092   5094#  5094   5096#  5096
        5098#  5098   5103#  5103   5105#  5105   5107#  5107   5112#  5112   5114#  5114   5116#  5116
        5328#  5328   5330#  5330   5332#  5332   5334#  5334   5336#  5336   5341#  5341   5343#  5343
        5345#  5345   5347#  5347   5349#  5349   5351#  5351   5353#  5353   5358#  5358   5360#  5360
        5362#  5362   5364#  5364   5369#  5369   5371#  5371   5376#  5376   5378#  5378   5380#  5380
        5385#  5385   5387#  5387   5389#  5389   5391#  5391   5393#  5393   5398#  5398   5400#  5400
        5402#  5402   5407#  5407   5409#  5409   5411#  5411
MWORD    198    200    202    204    209    211    213    215    220    222    224    226    231    233
         235    237    242    244    246    248    253    255    257    259    264    266    268    270
         275    277    279    281    283    285    287    391    393    395    397    402    404    406
         408    413    415    417    419    424    426    428    430    435    437    439    441    446
         448    450    452    457    459    461    463    468    470    472    474    476    478    480
         482    484    486    488    490    492    494    499    501    503    505    510    512    514
         516    521    523    525    527    532    534    536    538    543    545    547    549    554
         556    558    560    565    567    569    571    576    578    580    582    584    586    588
         590    592    594    596    598    600    602    607    609    611    613    615    617    619
         621    623    625    627    629    631    633    739    741    743    745    829    831    833
         835    919    921    923    925   1018   1020   1022   1024   1117   1119   1121   1123   1216
        1218   1220   1222   1315   1317   1319   1321   1416   1418   1420   1422   1577   1579   1581
        1584   1586   1588   1593   1595   1597   1600   1602   1604   1609   1611   1613   1616   1618
        1620   1625   1627   1629   1632   1634   1636   1641   1643   1645   1648   1650   1652   1657
        1659   1661   1664   1666   1668   1673   1675   1677   1680   1682   1684   1689   1691   1693
        1696   1698   1700   1705   1707   1709   1712   1714   1716   1721   1723   1725   1728   1730
        1732   1737   1739   1741   1744   1746   1748   1753   1755   1757   1760   1762   1764   1769
        1771   1773   1776   1778   1780   1785   1787   1789   1792   1794   1796   1801   1803   1805
        1808   1810   1812   1817   1819   1821   1824   1826   1828   1912   1914   1916   1919   1921
        1923   1925   2051   2053   2055   2057   2059   2061   2063   2065   2067   2069   2071   2073
        2075   2077   2079   2081   2083   2088   2090   2092   2094   2096   2098   2100   2102   2104
        2106   2108   2110   2112   2114   2116   2118   2120   2245   2247   2249   2251   2253   2255
        2257   2259   2261   2263   2265   2267   2269   2271   2273   2275   2277   2279   2281   2283
        2285   2287   2289   2291   2293   2295   2297   2299   2301   2303   2305   2307   2309   2311
        2313   2318   2320   2322   2324   2326   2328   2330   2332   2334   2336   2338   2340   2342
        2344   2346   2348   2350   2352   2357   2359   2361   2363   2365   2367   2369   2371   2373
        2375   2377   2379   2381   2475   2477   2479   2481   2483   2485   2487   2489   2491   2493
        2495   2497   2499   2501   2503   2505   2507   2509   2511   2513   2515   2517   2519   2521
        2523   2525   2527   2529   2531   2533   2535   2537   2539   2541   2543   2545   2550   2552
        2554   2556   2558   2560   2562   2564   2566   2568   2570   2572   2574   2576   2578   2580
        2582   2584   2586   2588   2593   2595   2597   2599   2601   2603   2605   2607   2609   2611
        2613   2615   2709   2711   2713   2715   2717   2719   2721   2723   2725   2727   2729   2731
        2733   2735   2737   2739   2741   2743   2745   2747   2749   2751   2753   2755   2757   2759
        2761   2763   2765   2767   2769   2771   2773   2775   2777   2779   2784   2786   2788   2790
        2792   2794   2796   2798   2800   2802   2804   2806   2808   2810   2812   2814   2816   2818
        2820   2822   2824   2826   2828   2830   2835   2837   2839   2841   2843   2845   2847   2849
        2851   2853   2855   2857   2859   2861   2955   2957   2959   2961   2963   2965   2967   2969
        2971   2973   2975   2977   2979   2981   2983   2985   2987   2989   2991   2993   2995   2997
        2999   3001   3003   3008   3010   3012   3014   3016   3018   3020   3022   3024   3026   3028
        3030   3032   3034   3036   3038   3040   3042   3044   3046   3048   3050   3052   3054   3056
        3058   3060   3062   3064   3066   3071   3073   3075   3077   3079   3081   3083   3085   3087
        3089   3091   3093   3095   3097   3099   3101   3103   3105   3107   3109   3111   3113   3115
        3117   3119   3121   3123   3125   3127   3129   3131   3133   3135   3137   3139   3141   3143
        3145   3147   3149   3243   3245   3247   3250   3252   3254   3256   3258   3260   3262   3264
```

```
            3266    3268    3270    3272    3274    3276    3278    3280    3282    3284    3286    3288    3290    3292      SEQ 1095
            3294    3296    3298    3300    3302    3304    3306    3308    3310    3312    3314    3316    3318    3320
            3325    3327    3329    3331    3333    3335    3337    3339    3341    3343    3345    3347    3349    3351
            3353    3355    3357    3359    3361    3363    3365    3367    3369    3371    3373    3375    3377    3379
            3384    3386    3388    3390    3392    3394    3396    3398    3400    3402    3404    3406    3408    3410
            3412    3414    3416    3418    3420    3422    3424    3426    3428    3430    3432    3434    3436    3438
            3440    3442    3444    3446    3448    3450    3452    3454    3456    3458    3460    3462    3464    3466
            3468    3470    3472    3474    3476    3478    3480    3482    3484    3486    3488    3490    3492    3494
            3496    3498    3500    3502    3504    3506    3508    3510    3512    3514    3516    3724    3726    3728
            3730    3732    3734    3736    3738    3740    3742    3747    3749    3751    3753    3755    3757    3759
            3761    3766    3768    3770    3772    3777    3779    3781    3783    3785    3787    3789    3791    3796
            3798    3800    3802    3804    3806    3808    3810    3812    3817    3819    3821    3823    3825    3830
            3832    3834    3839    3841    3843    4055    4057    4059    4061    4063    4065    4067    4069    4071
            4073    4078    4080    4082    4084    4086    4088    4090    4092    4097    4099    4101    4103    4108
            4110    4112    4114    4116    4118    4120    4122    4127    4129    4131    4133    4135    4137    4139
            4141    4143    4148    4150    4152    4154    4156    4161    4163    4165    4170    4172    4174    4385
            4387    4389    4391    4393    4395    4397    4402    4404    4406    4408    4410    4412    4414    4419
            4421    4423    4425    4430    4432    4434    4436    4438    4440    4442    4447    4449    4451    4453
            4455    4457    4459    4461    4466    4468    4470    4472    4474    4479    4481    4483    4488    4490
            4492    4717    4719    4721    4723    4725    4730    4732    4734    4736    4738    4740    4742    4747
            4749    4751    4753    4758    4760    4765    4767    4769    4774    4776    4778    4780    4782    4787
            4789    4791    4796    4798    4800    5009    5011    5013    5015    5017    5019    5021    5026    5028
            5030    5032    5034    5036    5038    5043    5045    5047    5049    5054    5056    5058    5060    5062
            506[illegible]    5066    5071    5073    5075    5077    5079    5081    5083    5085    5090    5092    5094    5096
            5098    5103    5105    5107    5112    5114    5116    5328    5330    5332    5334    5336    5341    5343
            5345    5347    5349    5351    5353    5358    5360    5362    5364    5369    5371    5376    5378    5380
            5385    5387    5389    5391    5393    5398    5400    5402    5407    5409    5411
PNTHW       5421    5424    5431    5437
PNTMSG      5419    5422    5427    5429    5433    5435
PUT         5452    5453
RGET        5464
RPUT        5452
RTN           88     113     331     356     370     380     689     720     787     812     877     902     969    1000
            1068    1099    1167    1198    1266    1297    1367    1398    1475    1500    1870    1895    1972    2003
            2195    2220    2431    2456    2665    2690    2911    2936    3199    3224    3568    3593    3617    3622
            3627    3632    3656    3686    3690    3699    3701    3706    3712    3894    3919    3943    3949    3955
            3960    3984    4018    4024    4031    4037    4043    4225    4250    4274    4279    4284    4289    4313
            4344    4350    4361    4366    4373    4543    4568    4592    4598    4604    4609    4633    4668    4670
            4679    4689    4694    4705    4851    4876    4900    4905    4910    4915    4939    4971    4977    4985
            4990    4997    5167    5192    5216    5221    5226    5231    5255    5286    5294    5304    5309    5318
            5438    5465
SCOPER       106     349     713     805     895     993    1092    1191    1290    1391    1493    1888    1996    2213
            2449    2683    2929    3217    3586    3912    4243    [illegible]    4869    5185
TMSG        5422
TMSGC       5419    5427    5429    5433    5435
```

```
                                         TITLE   DFPTA5
                                         SUBTTL  EBUS/MPROC 2901 Tests (Part 2)

                                         SALL

                                         SEARCH  DFPTA,MONSYM

                                 ; INTERN's

                                         ENTRY   TSTA31

                                 ; EXTERN's

                                         EXTERN  TSTA32,TSTA36,TSTA37,TSTA40,TSTA41

                                 ; EXTERN's located in DFPTA1.MAC (Mainline code..)

                                         EXTERN  TLOAD,TRACE,TSTSUB,AEXEC,AAPNT

                                 ; EXTERN's located in DFPTAI.MAC (Port Device Handling Routines module)

                                         EXTERN  IPACLR


                                 ;#**************************************************************************
                                 ; Z5 - Address for use in DDT
                                 ;#**************************************************************************

        000000'                  Z5:                                     ; address of 00000'

```

```
                                         SUBTTL  Register Interference Tests

                                 ;#**********************************************************************
                                 ;* Test 31 - 2901 Q-Register Interference Test
                                 ;
                                 ; Description:  Verify that the Q-Register is  not  loaded by any
                                 ;               destination functions other than 0, 4, and 6. And
                                 ;               that the contents of the Q-Register is unaffected
                                 ;               by any combination of source field, ALU  function
                                 ;               field, or destination field (except 0, 4, 6).
                                 ;
                                 ;               This includes  testing  when  the  Q-Register  is
                                 ;               being shifted left or right.
                                 ;
                                 ; Procedure:    Clear Port
                                 ;               Load microcode/set start address 0
                                 ;               Start microcode (will halt with a parity error at
                                 ;                 completion of test).
                                 ;
                                 ;               Execute  first  block of microinstructions to set
                                 ;               up non-zero  register contents in Registers 0-17.
                                 ;               This also sets up a error call subroutine.
                                 ;
                                 ;               Execute  the  next 640 microinstructions - if the
                                 ;               contents of the  Q-Register  ever  goes non-zero,
                                 ;               the microsequencer goes to the error subroutine.
                                 ;
                                 ;               When done, read the LAR and verify that microcode
                                 ;               stopped in the right place (location 2677),  with
                                 ;               a cram parity error.
                                 ;
                                 ;               If an error occurred, the microcode goes to  4000
                                 ;               and halts with a cram parity error.
                                 ;
                                 ; Failure:      ---
                                 ;#**********************************************************************

                                 ; Test data

000000' 254 00 0 00 000012'      TSTA31: JRST    TG31                     ; go start test
000001' 420402  000031                   EBUS!ALU!NDMP!ZALU!31            ; test mask
000002' 000033' 003702'                  T31M,,[ASCIZ ^2901 Q-Register Interference Test^]
000003' 003711' 000000                   [EXP MLAST!E23],,0
000004' 000000000000#                    TSTA32                           ; failure test table
000005' 000000000000#                    TSTA36                           ;   ...
000006' 000000000000#                    TSTA37
000007' 000000000000#                    TSTA40
000010' 000000000000#                    TSTA41
000011' 777777  777777                   -1
```

```

                                        ; Start test

000012' 201 00 0 00 000000'     TG31:   MOVEI   Z5                      ; get address of module start
000013' 260 17 0 00 000000*             GO      TRACE                   ; handle trace output
000014' 201 01 0 00 000033'             MOVEI   1,T31M                  ; set up microcode address
000015' 260 17 0 00 000000*             GO      TLOAD                   ; load/verify it
000016' 263 17 0 00 000000              RTN                             ; failed - exit test

                                ; 1st segment of test (Segment A) - Start up port and wait till it stops

000017' 400 15 0 00 000000      TA31:   SETZ    ERFLG,                  ; clear error flag
000020' 201 01 0 00 000012              MOVEI   1,^D10                  ; maximum delay before timeout (msec)
000021' 201 02 0 00 004000              MOVEI   2,4000                  ; error address
000022' 201 03 0 00 002677              MOVEI   3,2677                  ; correct address
000023' 400 04 0 00 000000              SETZ    4,                      ; starting address
000024' 260 17 0 00 000000*             GO      AEXEC                   ; execute ALU type test
000025' 474 15 0 00 000000              SETO    ERFLG,                  ; error occurred
000026' 027 00 0 00 000032'             SCOPER  MA31                    ; print error message
000027' 254 00 0 00 000017'             JRST    TA31                    ; loop on error
000030' 254 00 0 00 000031'             JRST    TX31                    ; altmode exit

                                ; End of test

000031' 263 17 0 00 000000      TX31:   RTN                             ; return

                                ; Error message - Microcode did not stop in the right place

000032' 000000000000#           MA31:   LAST!CALL!TXALL!AAPNT           ; print results of ALU type test
```

```
   109
   110                                         ; Microcode (979 words are loaded, 658 executed if the test passes)
   111
   112  000033'                                T31M:
   113
   114                                         ;       Clear Q Register
   115
   116  000033' 000000  010000                         MWORD   <ADDR=0,JMAP,J=1,B=0,SDO,AND,D=0>   ; 0
   117  000034' 740000  000040
   118
   119                                         ;       Initialize Registers 0-17 non-zero.
   120
   121  000035' 000100  000003                         MWORD   <CONT,B=1,SDO,OR,D=2,SKCN,MGC=3>    ; 1
   122  000036' 732000  640340
   123  000037' 000200  000004                         MWORD   <CONT,B=2,SDO,OR,D=2,SKCN,MGC=4>    ; 2
   124  000040' 732001  240340
   125  000041' 000300  000005                         MWORD   <CONT,B=3,SDO,OR,D=2,SKCN,MGC=5>    ; 3
   126  000042' 732001  640340
   127  000043' 000400  000006                         MWORD   <CONT,B=4,SDO,OR,D=2,SKCN,MGC=6>    ; 4
   128  000044' 732002  240340
   129  000045' 000500  000007                         MWORD   <CONT,B=5,SDO,OR,D=2,SKCN,MGC=7>    ; 5
   130  000046' 732002  640340
   131  000047' 000600  000010                         MWORD   <CONT,B=6,SDO,OR,D=2,SKCN,MGC=10>   ; 6
   132  000050' 732003  240340
   133  000051' 000700  000011                         MWORD   <CONT,B=7,SDO,OR,D=2,SKCN,MGC=11>   ; 7
   134  000052' 732003  640340
   135  000053' 001000  000012                         MWORD   <CONT,B=10,SDO,OR,D=2,SKCN,MGC=12>  ; 10
   136  000054' 732004  240340
   137  000055' 001100  000013                         MWORD   <CONT,B=11,SDO,OR,D=2,SKCN,MGC=13>  ; 11
   138  000056' 732004  640340
   139  000057' 001200  000014                         MWORD   <CONT,B=12,SDO,OR,D=2,SKCN,MGC=14>  ; 12
   140  000060' 732005  240340
   141  000061' 001300  000015                         MWORD   <CONT,B=13,SDO,OR,D=2,SKCN,MGC=15>  ; 13
   142  000062' 732005  640340
   143  000063' 001400  000016                         MWORD   <CONT,B=14,SDO,OR,D=2,SKCN,MGC=16>  ; 14
   144  000064' 732006  240340
   145  000065' 001500  000017                         MWORD   <CONT,B=15,SDO,OR,D=2,SKCN,MGC=17>  ; 15
   146  000066' 732006  640340
   147  000067' 001600  000020                         MWORD   <CONT,B=16,SDO,OR,D=2,SKCN,MGC=20>  ; 16
   148  000070' 732007  240340
   149  000071' 001700  000021                         MWORD   <CONT,B=17,SDO,OR,D=2,SKCN,MGC=21>  ; 17
   150  000072' 732007  640340
   151  000073' 002000  000022                         MWORD   <CONT,B=0,SDO,OR,D=2,SKCN,MGC=22>   ; 20
   152  000074' 732000  240340
   153  000075' 002110  000000                         MWORD   <JMAP,J=1000,D=1>                   ; 21 go test
   154  000076' 001000  000040
   155
   156                                         ;       Do the testing - do  something  that  should  not
   157                                         ;       affect the Q-Register, then verify the Q-Register
   158                                         ;       is still zero. If not go to 4000 which halts with
   159                                         ;       a parity error.
   160
   161  000077' 100000  000000                         MWORD   <ADDR=1000,CONT,B=5,SAQ,PLUS,D=1>   ; 1000
   162  000100' 001002  400340
   163  000101' 100110  030000                         MWORD   <CJV,J=1003,SOQ,OR,D=1,CENA,CCFZ>   ; 1001
```

```
000102' 231400  020140
000103' 100240  000000          MWORD   <JMAP,J=4000>                           ; 1002 error
000104' 000000  000040
000105' 100300  000000          MWORD   <CONT,A=1,B=6,SAQ,PLUS,D=2>             ; 1003
000106' 002013  000340
000107' 100410  060000          MWORD   <CJV,J=1006,SOQ,OR,D=1,CENA,CCFZ>       ; 1004
000110' 231400  020140
000111' 100540  000000          MWORD   <JMAP,J=4000>                           ; 1005 error
000112' 000000  000040
000113' 100600  000000          MWORD   <CONT,A=2,B=7,SAQ,PLUS,D=3>             ; 1006
000114' 003023  400340
000115' 100710  110000          MWORD   <CJV,J=1011,SOQ,OR,D=1,CENA,CCFZ>       ; 1007
000116' 231400  020140
000117' 101040  000000          MWORD   <JMAP,J=4000>                           ; 1010 error
000120' 000000  000040
000121' 101100  000000          MWORD   <CONT,A=3,B=10,SAQ,PLUS,D=5>            ; 1011
000122' 005034  000340
000123' 101210  140000          MWORD   <CJV,J=1014,SOQ,OR,D=1,CENA,CCFZ>       ; 1012
000124' 231400  020140
000125' 101340  000000          MWORD   <JMAP,J=4000>                           ; 1013 error
000126' 000000  000040
000127' 101400  000000          MWORD   <CONT,A=4,B=11,SAQ,PLUS,D=7>            ; 1014
000130' 007044  400340
000131' 101510  170000          MWORD   <CJV,J=1017,SOQ,OR,D=1,CENA,CCFZ>       ; 1015
000132' 231400  020140
000133' 101640  000000          MWORD   <JMAP,J=4000>                           ; 1016 error
000134' 000000  000040
000135' 101700  000000          MWORD   <CONT,A=12,B=17,SAB,PLUS,D=1>           ; 1017
000136' 101127  400340
000137' 102010  220000          MWORD   <CJV,J=1022,SOQ,OR,D=1,CENA,CCFZ>       ; 1020
000140' 231400  020140
000141' 102140  000000          MWORD   <JMAP,J=4000>                           ; 1021 error
000142' 000000  000040
000143' 102200  000000          MWORD   <CONT,A=13,B=0,SAB,PLUS,D=2>            ; 1022
000144' 102130  000340
000145' 102310  250000          MWORD   <CJV,J=1025,SOQ,OR,D=1,CENA,CCFZ>       ; 1023
000146' 231400  020140
000147' 102440  000000          MWORD   <JMAP,J=4000>                           ; 1024 error
000150' 000000  000040
000151' 102500  000000          MWORD   <CONT,A=14,B=1,SAB,PLUS,D=3>            ; 1025
000152' 103140  400340
000153' 102610  300000          MWORD   <CJV,J=1030,SOQ,OR,D=1,CENA,CCFZ>       ; 1026
000154' 231400  020140
000155' 102740  000000          MWORD   <JMAP,J=4000>                           ; 1027 error
000156' 000000  000040
000157' 103000  000000          MWORD   <CONT,A=15,B=2,SAB,PLUS,D=5>            ; 1030
000160' 105151  000340
000161' 103110  330000          MWORD   <CJV,J=1033,SOQ,OR,D=1,CENA,CCFZ>       ; 1031
000162' 231400  020140
000163' 103240  000000          MWORD   <JMAP,J=4000>                           ; 1032 error
000164' 000000  000040
000165' 103300  000000          MWORD   <CONT,A=16,B=3,SAB,PLUS,D=7>            ; 1033
000166' 107161  400340
000167' 103410  360000          MWORD   <CJV,J=1036,SOQ,OR,D=1,CENA,CCFZ>       ; 1034
000170' 231400  020140
```

```
000171' 103540  000000          MWORD   <JMAP,J=4000>                       ; 1035 error
000172' 000000  000040
000173' 103600  000000          MWORD   <CONT,A=4,B=11,SOQ,PLUS,D=1>        ; 1036
000174' 201044  400340
000175' 103710  410000          MWORD   <CJV,J=1041,SOQ,OR,D=1,CENA,CCFZ>   ; 1037
000176' 231400  020140
000177' 104040  000000          MWORD   <JMAP,J=4000>                       ; 1040 error
000200' 000000  000040
000201' 104100  000000          MWORD   <CONT,A=5,B=12,SOQ,PLUS,D=2>        ; 1041
000202' 202055  000340
000203' 104210  440000          MWORD   <CJV,J=1044,SOQ,OR,D=1,CENA,CCFZ>   ; 1042
000204' 231400  020140
000205' 104340  000000          MWORD   <JMAP,J=4000>                       ; 1043 error
000206' 000000  000040
000207' 104400  000000          MWORD   <CONT,A=6,B=13,SOQ,PLUS,D=3>        ; 1044
000210' 203065  400340
000211' 104510  470000          MWORD   <CJV,J=1047,SOQ,OR,D=1,CENA,CCFZ>   ; 1045
000212' 231400  020140
000213' 104640  000000          MWORD   <JMAP,J=4000>                       ; 1046 error
000214' 000000  000040
000215' 104700  000000          MWORD   <CONT,A=7,B=14,SOQ,PLUS,D=5>        ; 1047
000216' 205076  000340
000217' 105010  520000          MWORD   <CJV,J=1052,SOQ,OR,D=1,CENA,CCFZ>   ; 1050
000220' 231400  020140
000221' 105140  000000          MWORD   <JMAP,J=4000>                       ; 1051 error
000222' 000000  000040
000223' 105200  000000          MWORD   <CONT,A=10,B=15,SOQ,PLUS,D=7>       ; 1052
000224' 207106  400340
000225' 105310  550000          MWORD   <CJV,J=1055,SOQ,OR,D=1,CENA,CCFZ>   ; 1053
000226' 231400  020140
000227' 105440  000000          MWORD   <JMAP,J=4000>                       ; 1054 error
000230' 000000  000040
000231' 105500  000000          MWORD   <CONT,A=16,B=3,SOB,PLUS,D=1>        ; 1055
000232' 301161  400340
000233' 105610  600000          MWORD   <CJV,J=1060,SOQ,OR,D=1,CENA,CCFZ>   ; 1056
000234' 231400  020140
000235' 105740  000000          MWORD   <JMAP,J=4000>                       ; 1057 error
000236' 000000  000040
000237' 106000  000000          MWORD   <CONT,A=17,B=4,SOB,PLUS,D=2>        ; 1060
000240' 302172  000340
000241' 106110  630000          MWORD   <CJV,J=1063,SOQ,OR,D=1,CENA,CCFZ>   ; 1061
000242' 231400  020140
000243' 106240  000000          MWORD   <JMAP,J=4000>                       ; 1062 error
000244' 000000  000040
000245' 106300  000000          MWORD   <CONT,A=0,B=5,SOB,PLUS,D=3>         ; 1063
000246' 303002  400340
000247' 106410  660000          MWORD   <CJV,J=1066,SOQ,OR,D=1,CENA,CCFZ>   ; 1064
000250' 231400  020140
000251' 106540  000000          MWORD   <JMAP,J=4000>                       ; 1065 error
000252' 000000  000040
000253' 106600  000000          MWORD   <CONT,A=1,B=6,SOB,PLUS,D=5>         ; 1066
000254' 305013  000340
000255' 106710  710000          MWORD   <CJV,J=1071,SOQ,OR,D=1,CENA,CCFZ>   ; 1067
000256' 231400  020140
000257' 107040  000000          MWORD   <JMAP,J=4000>                       ; 1070 error
```

```
000260' 000000  000040
000261' 107100  000000          MWORD   <CONT,A=2,B=7,SOB,PLUS,D=7>             ; 1071
000262' 307023  400340
000263' 107210  740000          MWORD   <CJV,J=1074,SOQ,OR,D=1,CENA,CCFZ>       ; 1072
000264' 231400  020140
000265' 107340  000000          MWORD   <JMAP,J=4000>                           ; 1073 error
000266' 000000  000040
000267' 107400  000000          MWORD   <CONT,A=10,B=15,SOA,PLUS,D=1>           ; 1074
000270' 401106  400340
000271' 107510  770000          MWORD   <CJV,J=1077,SOQ,OR,D=1,CENA,CCFZ>       ; 1075
000272' 231400  020140
000273' 107640  000000          MWORD   <JMAP,J=4000>                           ; 1076 error
000274' 000000  000040
000275' 107700  000000          MWORD   <CONT,A=11,B=16,SOA,PLUS,D=2>           ; 1077
000276' 402117  000340
000277' 110011  020000          MWORD   <CJV,J=1102,SOQ,OR,D=1,CENA,CCFZ>       ; 1100
000300' 231400  020140
000301' 110140  000000          MWORD   <JMAP,J=4000>                           ; 1101 error
000302' 000000  000040
000303' 110200  000000          MWORD   <CONT,A=12,B=17,SOA,PLUS,D=3>           ; 1102
000304' 403127  400340
000305' 110311  050000          MWORD   <CJV,J=1105,SOQ,OR,D=1,CENA,CCFZ>       ; 1103
000306' 231400  020140
000307' 110440  000000          MWORD   <JMAP,J=4000>                           ; 1104 error
000310' 000000  000040
000311' 110500  000000          MWORD   <CONT,A=13,B=0,SOA,PLUS,D=5>            ; 1105
000312' 405130  000340
000313' 110611  100000          MWORD   <CJV,J=1110,SOQ,OR,D=1,CENA,CCFZ>       ; 1106
000314' 231400  020140
000315' 110740  000000          MWORD   <JMAP,J=4000>                           ; 1107 error
000316' 000000  000040
000317' 111000  000000          MWORD   <CONT,A=14,B=1,SOA,PLUS,D=7>            ; 1110
000320' 407140  400340
000321' 111111  130000          MWORD   <CJV,J=1113,SOQ,OR,D=1,CENA,CCFZ>       ; 1111
000322' 231400  020140
000323' 111240  000000          MWORD   <JMAP,J=4000>                           ; 1112 error
000324' 000000  000040
000325' 111300  000000          MWORD   <CONT,A=2,B=7,SDA,PLUS,D=1>             ; 1113
000326' 501023  400340
000327' 111411  160000          MWORD   <CJV,J=1116,SOQ,OR,D=1,CENA,CCFZ>       ; 1114
000330' 231400  020140
000331' 111540  000000          MWORD   <JMAP,J=4000>                           ; 1115 error
000332' 000000  000040
000333' 111600  000000          MWORD   <CONT,A=3,B=10,SDA,PLUS,D=2>            ; 1116
000334' 502034  000340
000335' 111711  210000          MWORD   <CJV,J=1121,SOQ,OR,D=1,CENA,CCFZ>       ; 1117
000336' 231400  020140
000337' 112040  000000          MWORD   <JMAP,J=4000>                           ; 1120 error
000340' 000000  000040
000341' 112100  000000          MWORD   <CONT,A=4,B=11,SDA,PLUS,D=3>            ; 1121
000342' 503044  400340
000343' 112211  240000          MWORD   <CJV,J=1124,SOQ,OR,D=1,CENA,CCFZ>       ; 1122
000344' 231400  020140
000345' 112340  000000          MWORD   <JMAP,J=4000>                           ; 1123 error
000346' 000000  000040
```

```
000347' 112400  000000          MWORD   <CONT,A=5,B=12,SDA,PLUS,D=5>            ; 1124
000350' 505055  000340
000351' 112511  270000          MWORD   <CJV,J=1127,SOQ,OR,D=1,CENA,CCFZ>       ; 1125
000352' 231400  020140
000353' 112640  000000          MWORD   <JMAP,J=4000>                           ; 1126 error
000354' 000000  000040
000355' 112700  000000          MWORD   <CONT,A=6,B=13,SDA,PLUS,D=7>            ; 1127
000356' 507065  400340
000357' 113011  320000          MWORD   <CJV,J=1132,SOQ,OR,D=1,CENA,CCFZ>       ; 1130
000360' 231400  020140
000361' 113140  000000          MWORD   <JMAP,J=4000>                           ; 1131 error
000362' 000000  000040
000363' 113200  000000          MWORD   <CONT,A=14,B=1,SDQ,PLUS,D=1>            ; 1132
000364' 601140  400340
000365' 113311  350000          MWORD   <CJV,J=1135,SOQ,OR,D=1,CENA,CCFZ>       ; 1133
000366' 231400  020140
000367' 113440  000000          MWORD   <JMAP,J=4000>                           ; 1134 error
000370' 000000  000040
000371' 113500  000000          MWORD   <CONT,A=15,B=2,SDQ,PLUS,D=2>            ; 1135
000372' 602151  000340
000373' 113611  400000          MWORD   <CJV,J=1140,SOQ,OR,D=1,CENA,CCFZ>       ; 1136
000374' 231400  020140
000375' 113740  000000          MWORD   <JMAP,J=4000>                           ; 1137 error
000376' 000000  000040
000377' 114000  000000          MWORD   <CONT,A=16,B=3,SDQ,PLUS,D=3>            ; 1140
000400' 603161  400340
000401' 114111  430000          MWORD   <CJV,J=1143,SOQ,OR,D=1,CENA,CCFZ>       ; 1141
000402' 231400  020140
000403' 114240  000000          MWORD   <JMAP,J=4000>                           ; 1142 error
000404' 000000  000040
000405' 114300  000000          MWORD   <CONT,A=17,B=4,SDQ,PLUS,D=5>            ; 1143
000406' 605172  000340
000407' 114411  460000          MWORD   <CJV,J=1146,SOQ,OR,D=1,CENA,CCFZ>       ; 1144
000410' 231400  020140
000411' 114540  000000          MWORD   <JMAP,J=4000>                           ; 1145 error
000412' 000000  000040
000413' 114600  000000          MWORD   <CONT,A=0,B=5,SDQ,PLUS,D=7>             ; 1146
000414' 607002  400340
000415' 114711  510000          MWORD   <CJV,J=1151,SOQ,OR,D=1,CENA,CCFZ>       ; 1147
000416' 231400  020140
000417' 115040  000000          MWORD   <JMAP,J=4000>                           ; 1150 error
000420' 000000  000040
000421' 115100  000000          MWORD   <CONT,A=6,B=13,SDO,PLUS,D=1>            ; 1151
000422' 701065  400340
000423' 115211  540000          MWORD   <CJV,J=1154,SOQ,OR,D=1,CENA,CCFZ>       ; 1152
000424' 231400  020140
000425' 115340  000000          MWORD   <JMAP,J=4000>                           ; 1153 error
000426' 000000  000040
000427' 115400  000000          MWORD   <CONT,A=7,B=14,SDO,PLUS,D=2>            ; 1154
000430' 702076  000340
000431' 115511  570000          MWORD   <CJV,J=1157,SOQ,OR,D=1,CENA,CCFZ>       ; 1155
000432' 231400  020140
000433' 115640  000000          MWORD   <JMAP,J=4000>                           ; 1156 error
000434' 000000  000040
000435' 115700  000000          MWORD   <CONT,A=10,B=15,SDO,PLUS,D=3>           ; 1157
```

```
000436' 703106  400340
000437' 116011  620000          MWORD   <CJV,J=1162,SOQ,OR,D=1,CENA,CCFZ>   ; 1160
000440' 231400  020140
000441' 116140  000000          MWORD   <JMAP,J=4000>                       ; 1161 error
000442' 000000  000040
000443' 116200  000000          MWORD   <CONT,A=11,B=16,SD0,PLUS,D=5>       ; 1162
000444' 705117  000340
000445' 116311  650000          MWORD   <CJV,J=1165,SOQ,OR,D=1,CENA,CCFZ>   ; 1163
000446' 231400  020140
000447' 116440  000000          MWORD   <JMAP,J=4000>                       ; 1164 error
000450' 000000  000040
000451' 116500  000000          MWORD   <CONT,A=12,B=17,SD0,PLUS,D=7>       ; 1165
000452' 707127  400340
000453' 116611  700000          MWORD   <CJV,J=1170,SOQ,OR,D=1,CENA,CCFZ>   ; 1166
000454' 231400  020140
000455' 116740  000000          MWORD   <JMAP,J=4000>                       ; 1167 error
000456' 000000  000040
000457' 117000  000000          MWORD   <CONT,A=0,B=5,SAQ,SMIN,D=1>         ; 1170
000460' 011002  400340
000461' 117111  730000          MWORD   <CJV,J=1173,SOQ,OR,D=1,CENA,CCFZ>   ; 1171
000462' 231400  020140
000463' 117240  000000          MWORD   <JMAP,J=4000>                       ; 1172 error
000464' 000000  000040
000465' 117300  000000          MWORD   <CONT,A=1,B=6,SAQ,SMIN,D=2>         ; 1173
000466' 012013  000340
000467' 117411  760000          MWORD   <CJV,J=1176,SOQ,OR,D=1,CENA,CCFZ>   ; 1174
000470' 231400  020140
000471' 117540  000000          MWORD   <JMAP,J=4000>                       ; 1175 error
000472' 000000  000040
000473' 117600  000000          MWORD   <CONT,A=2,B=7,SAQ,SMIN,D=3>         ; 1176
000474' 013023  400340
000475' 117712  010000          MWORD   <CJV,J=1201,SOQ,OR,D=1,CENA,CCFZ>   ; 1177
000476' 231400  020140
000477' 120040  000000          MWORD   <JMAP,J=4000>                       ; 1200 error
000500' 000000  000040
000501' 120100  000000          MWORD   <CONT,A=3,B=10,SAQ,SMIN,D=5>        ; 1201
000502' 015034  000340
000503' 120212  040000          MWORD   <CJV,J=1204,SOQ,OR,D=1,CENA,CCFZ>   ; 1202
000504' 231400  020140
000505' 120340  000000          MWORD   <JMAP,J=4000>                       ; 1203 error
000506' 000000  000040
000507' 120400  000000          MWORD   <CONT,A=4,B=11,SAQ,SMIN,D=7>        ; 1204
000510' 017044  400340
000511' 120512  070000          MWORD   <CJV,J=1207,SOQ,OR,D=1,CENA,CCFZ>   ; 1205
000512' 231400  020140
000513' 120640  000000          MWORD   <JMAP,J=4000>                       ; 1206 error
000514' 000000  000040
000515' 120700  000000          MWORD   <CONT,A=12,B=17,SAB,SMIN,D=1>       ; 1207
000516' 111127  400340
000517' 121012  120000          MWORD   <CJV,J=1212,SOQ,OR,D=1,CENA,CCFZ>   ; 1210
000520' 231400  020140
000521' 121140  000000          MWORD   <JMAP,J=4000>                       ; 1211 error
000522' 000000  000040
000523' 121200  000000          MWORD   <CONT,A=13,B=0,SAB,SMIN.D=2>        ; 1212
000524' 112130  000340
```

```
 439  000525' 121312  150000              MWORD   <CJV,J=1215,SOQ,OR,D=1,CENA,CCFZ>    ; 1213
 440  000526' 231400  020140
 441  000527' 121440  000000              MWORD   <JMAP,J=4000>                        ; 1214 error
 442  000530' 000000  000040
 443  000531' 121500  000000              MWORD   <CONT,A=14,B=1,SAB,SMIN,D=3>         ; 1215
 444  000532' 113140  400340
 445  000533' 121612  200000              MWORD   <CJV,J=1220,SOQ,OR,D=1,CENA,CCFZ>    ; 1216
 446  000534' 231400  020140
 447  000535' 121740  000000              MWORD   <JMAP,J=4000>                        ; 1217 error
 448  000536' 000000  000040
 449  000537' 122000  000000              MWORD   <CONT,A=15,B=2,SAB,SMIN,D=5>         ; 1220
 450  000540' 115151  000340
 451  000541' 122112  230000              MWORD   <CJV,J=1223,SOQ,OR,D=1,CENA,CCFZ>    ; 1221
 452  000542' 231400  020140
 453  000543' 122240  000000              MWORD   <JMAP,J=4000>                        ; 1222 error
 454  000544' 000000  000040
 455  000545' 122300  000000              MWORD   <CONT,A=16,B=3,SAB,SMIN,D=7>         ; 1223
 456  000546' 117161  400340
 457  000547' 122412  260000              MWORD   <CJV,J=1226,SOQ,OR,D=1,CENA,CCFZ>    ; 1224
 458  000550' 231400  020140
 459  000551' 122540  000000              MWORD   <JMAP,J=4000>                        ; 1225 error
 460  000552' 000000  000040
 461  000553' 122600  000000              MWORD   <CONT,A=4,B=11,SOQ,SMIN,D=1>         ; 1226
 462  000554' 211044  400340
 463  000555' 122712  310000              MWORD   <CJV,J=1231,SOQ,OR,D=1,CENA,CCFZ>    ; 1227
 464  000556' 231400  020140
 465  000557' 123040  000000              MWORD   <JMAP,J=4000>                        ; 1230 error
 466  000560' 000000  000040
 467  000561' 123100  000000              MWORD   <CONT,A=5,B=12,SOQ,SMIN,D=2>         ; 1231
 468  000562' 212055  000340
 469  000563' 123212  340000              MWORD   <CJV,J=1234,SOQ,OR,D=1,CENA,CCFZ>    ; 1232
 470  000564' 231400  020140
 471  000565' 123340  000000              MWORD   <JMAP,J=4000>                        ; 1233 error
 472  000566' 000000  000040
 473  000567' 123400  000000              MWORD   <CONT,A=6,B=13,SOQ,SMIN,D=3>         ; 1234
 474  000570' 213065  400340
 475  000571' 123512  370000              MWORD   <CJV,J=1237,SOQ,OR,D=1,CENA,CCFZ>    ; 1235
 476  000572' 231400  020140
 477  000573' 123640  000000              MWORD   <JMAP,J=4000>                        ; 1236 error
 478  000574' 000000  000040
 479  000575' 123700  000000              MWORD   <CONT,A=7,B=14,SOQ,SMIN,D=5>         ; 1237
 480  000576' 215076  000340
 481  000577' 124012  420000              MWORD   <CJV,J=1242,SOQ,OR,D=1,CENA,CCFZ>    ; 1240
 482  000600' 231400  020140
 483  000601' 124140  000000              MWORD   <JMAP,J=4000>                        ; 1241 error
 484  000602' 000000  000040
 485  000603' 124200  000000              MWORD   <CONT,A=10,B=15,SOQ,SMIN,D=7>        ; 1242
 486  000604' 217106  400340
 487  000605' 124312  450000              MWORD   <CJV,J=1245,SOQ,OR,D=1,CENA,CCFZ>    ; 1243
 488  000606' 231400  020140
 489  000607' 124440  000000              MWORD   <JMAP,J=4000>                        ; 1244 error
 490  000610' 000000  000040
 491  000611' 124500  000000              MWORD   <CONT,A=16,B=3,SOB,SMIN,D=1>         ; 1245
 492  000612' 311161  400340
 493  000613' 124612  500000              MWORD   <CJV,J=1250,SOQ,OR,D=1,CENA,CCFZ>    ; 1246
```

```
000614' 231400  020140
000615' 124740  000000          MWORD   <JMAP,J=4000>                           ; 1247 error
000616' 000000  000040
000617' 125000  000000          MWORD   <CONT,A=17,B=4,SOB,SMIN,D=2>            ; 1250
000620' 312172  000340
000621' 125112  530000          MWORD   <CJV,J=1253,SOQ,OR,D=1,CENA,CCFZ>       ; 1251
000622' 231400  020140
000623' 125240  000000          MWORD   <JMAP,J=4000>                           ; 1252 error
000624' 000000  000040
000625' 125300  000000          MWORD   <CONT,A=0,B=5,SOB,SMIN,D=3>             ; 1253
000626' 313002  400340
000627' 125412  560000          MWORD   <CJV,J=1256,SOQ,OR,D=1,CENA,CCFZ>       ; 1254
000630' 231400  020140
000631' 125540  000000          MWORD   <JMAP,J=4000>                           ; 1255 error
000632' 000000  000040
000633' 125600  000000          MWORD   <CONT,A=1,B=6,SOB,SMIN,D=5>             ; 1256
000634' 315013  000340
000635' 125712  610000          MWORD   <CJV,J=1261,SOQ,OR,D=1,CENA,CCFZ>       ; 1257
000636' 231400  020140
000637' 126040  000000          MWORD   <JMAP,J=4000>                           ; 1260 error
000640' 000000  000040
000641' 126100  000000          MWORD   <CONT,A=2,B=7,SOB,SMIN,D=7>             ; 1261
000642' 317023  400340
000643' 126212  640000          MWORD   <CJV,J=1264,SOQ,OR,D=1,CENA,CCFZ>       ; 1262
000644' 231400  020140
000645' 126340  000000          MWORD   <JMAP,J=4000>                           ; 1263 error
000646' 000000  000040
000647' 126400  000000          MWORD   <CONT,A=10,B=15,SOA,SMIN,D=1>           ; 1264
000650' 411106  400340
000651' 126512  670000          MWORD   <CJV,J=1267,SOQ,OR,D=1,CENA,CCFZ>       ; 1265
000652' 231400  020140
000653' 126640  000000          MWORD   <JMAP,J=4000>                           ; 1266 error
000654' 000000  000040
000655' 126700  000000          MWORD   <CONT,A=11,B=16,SOA,SMIN,D=2>           ; 1267
000656' 412117  000340
000657' 127012  720000          MWORD   <CJV,J=1272,SOQ,OR,D=1,CENA,CCFZ>       ; 1270
000660' 231400  020140
000661' 127140  000000          MWORD   <JMAP,J=4000>                           ; 1271 error
000662' 000000  000040
000663' 127200  000000          MWORD   <CONT,A=12,B=17,SOA,SMIN,D=3>           ; 1272
000664' 413127  400340
000665' 127312  750000          MWORD   <CJV,J=1275,SOQ,OR,D=1,CENA,CCFZ>       ; 1273
000666' 231400  020140
000667' 127440  000000          MWORD   <JMAP,J=4000>                           ; 1274 error
000670' 000000  000040
000671' 127500  000000          MWORD   <CONT,A=13,B=0,SOA,SMIN,D=5>            ; 1275
000672' 415130  000340
000673' 127613  000000          MWORD   <CJV,J=1300,SOQ,OR,D=1,CENA,CCFZ>       ; 1276
000674' 231400  020140
000675' 127740  000000          MWORD   <JMAP,J=4000>                           ; 1277 error
000676' 000000  000040
000677' 130000  000000          MWORD   <CONT,A=14,B=1,SOA,SMIN,D=7>            ; 1300
000700' 417140  400340
000701' 130113  030000          MWORD   <CJV,J=1303,SOQ,OR,D=1,CENA,CCFZ>       ; 1301
000702' 231400  020140
```

```
000703' 130240  000000          MWORD   <JMAP,J=4000>                           ; 1302 error
000704' 000000  000040
000705' 130300  000000          MWORD   <CONT,A=2,B=7,SDA,SMIN,D=1>             ; 1303
000706' 511023  400340
000707' 130413  060000          MWORD   <CJV,J=1306,SOQ,OR,D=1,CENA,CCFZ>       ; 1304
000710' 231400  020140
000711' 130540  000000          MWORD   <JMAP,J=4000>                           ; 1305 error
000712' 000000  000040
000713' 130600  000000          MWORD   <CONT,A=3,B=10,SDA,SMIN,D=2>            ; 1306
000714' 512034  000340
000715' 130713  110000          MWORD   <CJV,J=1311,SOQ,OR,D=1,CENA,CCFZ>       ; 1307
000716' 231400  020140
000717' 131040  000000          MWORD   <JMAP,J=4000>                           ; 1310 error
000720' 000000  000040
000721' 131100  000000          MWORD   <CONT,A=4,B=11,SDA,SMIN,D=3>            ; 1311
000722' 513044  400340
000723' 131213  140000          MWORD   <CJV,J=1314,SOQ,OR,D=1,CENA,CCFZ>       ; 1312
000724' 231400  020140
000725' 131340  000000          MWORD   <JMAP,J=4000>                           ; 1313 error
000726' 000000  000040
000727' 131400  000000          MWORD   <CONT,A=5,B=12,SDA,SMIN,D=5>            ; 1314
000730' 515055  000340
000731' 131513  170000          MWORD   <CJV,J=1317,SOQ,OR,D=1,CENA,CCFZ>       ; 1315
000732' 231400  020140
000733' 131640  000000          MWORD   <JMAP,J=4000>                           ; 1316 error
000734' 000000  000040
000735' 131700  000000          MWORD   <CONT,A=6,B=13,SDA,SMIN,D=7>            ; 1317
000736' 517065  400340
000737' 132013  220000          MWORD   <CJV,J=1322,SOQ,OR,D=1,CENA,CCFZ>       ; 1320
000740' 231400  020140
000741' 132140  000000          MWORD   <JMAP,J=4000>                           ; 1321 error
000742' 000000  000040
000743' 132200  000000          MWORD   <CONT,A=14,B=1,SDQ,SMIN,D=1>            ; 1322
000744' 611140  400340
000745' 132313  250000          MWORD   <CJV,J=1325,SOQ,OR,D=1,CENA,CCFZ>       ; 1323
000746' 231400  020140
000747' 132440  000000          MWORD   <JMAP,J=4000>                           ; 1324 error
000750' 000000  000040
000751' 132500  000000          MWORD   <CONT,A=15,B=2,SDQ,SMIN,D=2>            ; 1325
000752' 612151  000340
000753' 132613  300000          MWORD   <CJV,J=1330,SOQ,OR,D=1,CENA,CCFZ>       ; 1326
000754' 231400  020140
000755' 132740  000000          MWORD   <JMAP,J=4000>                           ; 1327 error
000756' 000000  000040
000757' 133000  000000          MWORD   <CONT,A=16,B=3,SDQ,SMIN,D=3>            ; 1330
000760' 613161  400340
000761' 133113  330000          MWORD   <CJV,J=1333,SOQ,OR,D=1,CENA,CCFZ>       ; 1331
000762' 231400  020140
000763' 133240  000000          MWORD   <JMAP,J=4000>                           ; 1332 error
000764' 000000  000040
000765' 133300  000000          MWORD   <CONT,A=17,B=4,SDQ,SMIN,D=5>            ; 1333
000766' 615172  000340
000767' 133413  360000          MWORD   <CJV,J=1336,SOQ,OR,D=1,CENA,CCFZ>       ; 1334
000770' 231400  020140
000771' 133540  000000          MWORD   <JMAP,J=4000>                           ; 1335 error
```

```
000772' 000000  000040
000773' 133600  000000          MWORD   <CONT,A=0,B=5,SDQ,SMIN,D=7>             ; 1336
000774' 617002  400340
000775' 133713  410000          MWORD   <CJV,J=1341,SOQ,OR,D=1,CENA,CCFZ>       ; 1337
000776' 231400  020140
000777' 134040  000000          MWORD   <JMAP,J=4000>                           ; 1340 error
001000' 000000  000040
001001' 134100  000000          MWORD   <CONT,A=6,B=13,SDO,SMIN,D=1>            ; 1341
001002' 711065  400340
001003' 134213  440000          MWORD   <CJV,J=1344,SOQ,OR,D=1,CENA,CCFZ>       ; 1342
001004' 231400  020140
001005' 134340  000000          MWORD   <JMAP,J=4000>                           ; 1343 error
001006' 000000  000040
001007' 134400  000000          MWORD   <CONT,A=7,B=14,SDO,SMIN,D=2>            ; 1344
001010' 712076  000340
001011' 134513  470000          MWORD   <CJV,J=1347,SOQ,OR,D=1,CENA,CCFZ>       ; 1345
001012' 231400  020140
001013' 134640  000000          MWORD   <JMAP,J=4000>                           ; 1346 error
001014' 000000  000040
001015' 134700  000000          MWORD   <CONT,A=10,B=15,SDO,SMIN,D=3>           ; 1347
001016' 713106  400340
001017' 135013  520000          MWORD   <CJV,J=1352,SOQ,OR,D=1,CENA,CCFZ>       ; 1350
001020' 231400  020140
001021' 135140  000000          MWORD   <JMAP,J=4000>                           ; 1351 error
001022' 000000  000040
001023' 135200  000000          MWORD   <CONT,A=11,B=16,SDO,SMIN,D=5>           ; 1352
001024' 715117  000340
001025' 135313  550000          MWORD   <CJV,J=1355,SOQ,OR,D=1,CENA,CCFZ>       ; 1353
001026' 231400  020140
001027' 135440  000000          MWORD   <JMAP,J=4000>                           ; 1354 error
001030' 000000  000040
001031' 135500  000000          MWORD   <CONT,A=12,B=17,SDO,SMIN,D=7>           ; 1355
001032' 717127  400340
001033' 135613  600000          MWORD   <CJV,J=1360,SOQ,OR,D=1,CENA,CCFZ>       ; 1356
001034' 231400  020140
001035' 135740  000000          MWORD   <JMAP,J=4000>                           ; 1357 error
001036' 000000  000040
001037' 136000  000000          MWORD   <CONT,A=0,B=5,SAQ,RMIN,D=1>             ; 1360
001040' 021002  400340
001041' 136113  630000          MWORD   <CJV,J=1363,SOQ,OR,D=1,CENA,CCFZ>       ; 1361
001042' 231400  020140
001043' 136240  000000          MWORD   <JMAP,J=4000>                           ; 1362 error
001044' 000000  000040
001045' 136300  000000          MWORD   <CONT,A=1,B=6,SAQ,RMIN,D=2>             ; 1363
001046' 022013  000340
001047' 136413  660000          MWORD   <CJV,J=1366,SOQ,OR,D=1,CENA,CCFZ>       ; 1364
001050' 231400  020140
001051' 136540  000000          MWORD   <JMAP,J=4000>                           ; 1365 error
001052' 000000  000040
001053' 136600  000000          MWORD   <CONT,A=2,B=7,SAQ,RMIN,D=3>             ; 1366
001054' 023023  400340
001055' 136713  710000          MWORD   <CJV,J=1371,SOQ,OR,D=1,CENA,CCFZ>       ; 1367
001056' 231400  020140
001057' 137040  000000          MWORD   <JMAP,J=4000>                           ; 1370 error
001060' 000000  000040
```

```
001061' 137100  000000          MWORD   <CONT,A=3,B=10,SAQ,RMIN,D=5>            ; 1371
001062' 025034  000340
001063' 137213  740000          MWORD   <CJV,J=1374,SOQ,OR,D=1,CENA,CCFZ>       ; 1372
001064' 231400  020140
001065' 137340  000000          MWORD   <JMAP,J=4000>                           ; 1373 error
001066' 000000  000040
001067' 137400  000000          MWORD   <CONT,A=4,B=11,SAQ,RMIN,D=7>            ; 1374
001070' 027044  400340
001071' 137513  770000          MWORD   <CJV,J=1377,SOQ,OR,D=1,CENA,CCFZ>       ; 1375
001072' 231400  020140
001073' 137640  000000          MWORD   <JMAP,J=4000>                           ; 1376 error
001074' 000000  000040
001075' 137700  000000          MWORD   <CONT,A=12,B=17,SAB,RMIN,D=1>           ; 1377
001076' 121127  400340
001077' 140014  020000          MWORD   <CJV,J=1402,SOQ,OR,D=1,CENA,CCFZ>       ; 1400
001100' 231400  020140
001101' 140140  000000          MWORD   <JMAP,J=4000>                           ; 1401 error
001102' 000000  000040
001103' 140200  000000          MWORD   <CONT,A=13,B=0,SAB,RMIN,D=2>            ; 1402
001104' 122130  000340
001105' 140314  050000          MWORD   <CJV,J=1405,SOQ,OR,D=1,CENA,CCFZ>       ; 1403
001106' 231400  020140
001107' 140440  000000          MWORD   <JMAP,J=4000>                           ; 1404 error
001110' 000000  000040
001111' 140500  000000          MWORD   <CONT,A=14,B=1,SAB,RMIN,D=3>            ; 1405
001112' 123140  400340
001113' 140614  100000          MWORD   <CJV,J=1410,SOQ,OR,D=1,CENA,CCFZ>       ; 1406
001114' 231400  020140
001115' 140740  000000          MWORD   <JMAP,J=4000>                           ; 1407 error
001116' 000000  000040
001117' 141000  000000          MWORD   <CONT,A=15,B=2,SAB,RMIN,D=5>            ; 1410
001120' 125151  000340
001121' 141114  130000          MWORD   <CJV,J=1413,SOQ,OR,D=1,CENA,CCFZ>       ; 1411
001122' 231400  020140
001123' 141240  000000          MWORD   <JMAP,J=4000>                           ; 1412 error
001124' 000000  000040
001125' 141300  000000          MWORD   <CONT,A=16,B=3,SAB,RMIN,D=7>            ; 1413
001126' 127161  400340
001127' 141414  160000          MWORD   <CJV,J=1416,SOQ,OR,D=1,CENA,CCFZ>       ; 1414
001130' 231400  020140
001131' 141540  000000          MWORD   <JMAP,J=4000>                           ; 1415 error
001132' 000000  000040
001133' 141600  000000          MWORD   <CONT,A=4,B=11,SOQ,RMIN,D=1>            ; 1416
001134' 221044  400340
001135' 141714  210000          MWORD   <CJV,J=1421,SOQ,OR,D=1,CENA,CCFZ>       ; 1417
001136' 231400  020140
001137' 142040  000000          MWORD   <JMAP,J=4000>                           ; 1420 error
001140' 000000  000040
001141' 142100  000000          MWORD   <CONT,A=5,B=12,SOQ,RMIN,D=2>            ; 1421
001142' 222055  000340
001143' 142214  240000          MWORD   <CJV,J=1424,SOQ,OR,D=1,CENA,CCFZ>       ; 1422
001144' 231400  020140
001145' 142340  000000          MWORD   <JMAP,J=4000>                           ; 1423 error
001146' 000000  000040
001147' 142400  000000          MWORD   <CONT,A=6,B=13,SOQ,RMIN,D=3>            ; 1424
```

```
001150' 223065  400340
001151' 142514  270000          MWORD   <CJV,J=1427,SOQ,OR,D=1,CENA,CCFZ>       ; 1425
001152' 231400  020140
001153' 142640  000000          MWORD   <JMAP,J=4000>                           ; 1426 error
001154' 000000  000040
001155' 142700  000000          MWORD   <CONT,A=7,B=14,SOQ,RMIN,D=5>            ; 1427
001156' 225076  000340
001157' 143014  320000          MWORD   <CJV,J=1432,SOQ,OR,D=1,CENA,CCFZ>       ; 1430
001160' 231400  020140
001161' 143140  000000          MWORD   <JMAP,J=4000>                           ; 1431 error
001162' 000000  000040
001163' 143200  000000          MWORD   <CONT,A=10,B=15,SOQ,RMIN,D=7>           ; 1432
001164' 227106  400340
001165' 143314  350000          MWORD   <CJV,J=1435,SOQ,OR,D=1,CENA,CCFZ>       ; 1433
001166' 231400  020140
001167' 143440  000000          MWORD   <JMAP,J=4000>                           ; 1434 error
001170' 000000  000040
001171' 143500  000000          MWORD   <CONT,A=16,B=3,SOB,RMIN,D=1>            ; 1435
001172' 321161  400340
001173' 143614  400000          MWORD   <CJV,J=1440,SOQ,OR,D=1,CENA,CCFZ>       ; 1436
001174' 231400  020140
001175' 143740  000000          MWORD   <JMAP,J=4000>                           ; 1437 error
001176' 000000  000040
001177' 144000  000000          MWORD   <CONT,A=17,B=4,SOB,RMIN,D=2>            ; 1440
001200' 322172  000340
001201' 144114  430000          MWORD   <CJV,J=1443,SOQ,OR,D=1,CENA,CCFZ>       ; 1441
001202' 231400  020140
001203' 144240  000000          MWORD   <JMAP,J=4000>                           ; 1442 error
001204' 000000  000040
001205' 144300  000000          MWORD   <CONT,A=0,B=5,SOB,RMIN,D=3>             ; 1443
001206' 323002  400340
001207' 144414  460000          MWORD   <CJV,J=1446,SOQ,OR,D=1,CENA,CCFZ>       ; 1444
001210' 231400  020140
001211' 144540  000000          MWORD   <JMAP,J=4000>                           ; 1445 error
001212' 000000  000040
001213' 144600  000000          MWORD   <CONT,A=1,B=6,SOB,RMIN,D=5>             ; 1446
001214' 325013  000340
001215' 144714  510000          MWORD   <CJV,J=1451,SOQ,OR,D=1,CENA,CCFZ>       ; 1447
001216' 231400  020140
001217' 145040  000000          MWORD   <JMAP,J=4000>                           ; 1450 error
001220' 000000  000040
001221' 145100  000000          MWORD   <CONT,A=2,B=7,SOB,RMIN,D=7>             ; 1451
001222' 327023  400340
001223' 145214  540000          MWORD   <CJV,J=1454,SOQ,OR,D=1,CENA,CCFZ>       ; 1452
001224' 231400  020140
001225' 145340  000000          MWORD   <JMAP,J=4000>                           ; 1453 error
001226' 000000  000040
001227' 145400  000000          MWORD   <CONT,A=10,B=15,SOA,RMIN,D=1>           ; 1454
001230' 421106  400340
001231' 145514  570000          MWORD   <CJV,J=1457,SOQ,OR,D=1,CENA,CCFZ>       ; 1455
001232' 231400  020140
001233' 145640  000000          MWORD   <JMAP,J=4000>                           ; 1456 error
001234' 000000  000040
001235' 145700  000000          MWORD   <CONT,A=11,B=16,SOA,RMIN,D=2>           ; 1457
001236' 422117  000340
```

```
001237' 146014  620000          MWORD   <CJV,J=1462,SOQ,OR,D=1,CENA,CCFZ>   ; 1460
001240' 231400  020140
001241' 146140  000000          MWORD   <JMAP,J=4000>                        ; 1461 error
001242' 000000  000040
001243' 146200  000000          MWORD   <CONT,A=12,B=17,SOA,RMIN,D=3>        ; 1462
001244' 423127  400340
001245' 146314  650000          MWORD   <CJV,J=1465,SOQ,OR,D=1,CENA,CCFZ>   ; 1463
001246' 231400  020140
001247' 146440  000000          MWORD   <JMAP,J=4000>                        ; 1464 error
001250' 000000  000040
001251' 146500  000000          MWORD   <CONT,A=13,B=0,SOA,RMIN,D=5>         ; 1465
001252' 425130  000340
001253' 146614  700000          MWORD   <CJV,J=1470,SOQ,OR,D=1,CENA,CCFZ>   ; 1466
001254' 231400  020140
001255' 146740  000000          MWORD   <JMAP,J=4000>                        ; 1467 error
001256' 000000  000040
001257' 147000  000000          MWORD   <CONT,A=14,B=1,SOA,RMIN,D=7>         ; 1470
001260' 427140  400340
001261' 147114  730000          MWORD   <CJV,J=1473,SOQ,OR,D=1,CENA,CCFZ>   ; 1471
001262' 231400  020140
001263' 147240  000000          MWORD   <JMAP,J=4000>                        ; 1472 error
001264' 000000  000040
001265' 147300  000000          MWORD   <CONT,A=2,B=7,SDA,RMIN,D=1>          ; 1473
001266' 521023  400340
001267' 147414  760000          MWORD   <CJV,J=1476,SOQ,OR,D=1,CENA,CCFZ>   ; 1474
001270' 231400  020140
001271' 147540  000000          MWORD   <JMAP,J=4000>                        ; 1475 error
001272' 000000  000040
001273' 147600  000000          MWORD   <CONT,A=3,B=10,SDA,RMIN,D=2>         ; 1476
001274' 522034  000340
001275' 147715  010000          MWORD   <CJV,J=1501,SOQ,OR,D=1,CENA,CCFZ>   ; 1477
001276' 231400  020140
001277' 150040  000000          MWORD   <JMAP,J=4000>                        ; 1500 error
001300' 000000  000040
001301' 150100  000000          MWORD   <CONT,A=4,B=11,SDA,RMIN,D=3>         ; 1501
001302' 523044  400340
001303' 150215  040000          MWORD   <CJV,J=1504,SOQ,OR,D=1,CENA,CCFZ>   ; 1502
001304' 231400  020140
001305' 150340  000000          MWORD   <JMAP,J=4000>                        ; 1503 error
001306' 000000  000040
001307' 150400  000000          MWORD   <CONT,A=5,B=12,SDA,RMIN,D=5>         ; 1504
001310' 525055  000340
001311' 150515  070000          MWORD   <CJV,J=1507,SOQ,OR,D=1,CENA,CCFZ>   ; 1505
001312' 231400  020140
001313' 150640  000000          MWORD   <JMAP,J=4000>                        ; 1506 error
001314' 000000  000040
001315' 150700  000000          MWORD   <CONT,A=6,B=13,SDA,RMIN,D=7>         ; 1507
001316' 527065  400340
001317' 151015  120000          MWORD   <CJV,J=1512,SOQ,OR,D=1,CENA,CCFZ>   ; 1510
001320' 231400  020140
001321' 151140  000000          MWORD   <JMAP,J=4000>                        ; 1511 error
001322' 000000  000040
001323' 151200  000000          MWORD   <CONT,A=14,B=1,SDQ,RMIN,D=1>         ; 1512
001324' 621140  400340
001325' 151315  150000          MWORD   <CJV,J=1515,SOQ,OR,D=1,CENA,CCFZ>   ; 1513
```

```
 824  001326' 231400  020140
 825  001327' 151440  000000            MWORD   <JMAP,J=4000>                          ; 1514 error
 826  001330' 000000  000040
 827  001331' 151500  000000            MWORD   <CONT,A=15,B=2,SDQ,RMIN,D=2>           ; 1515
 828  001332' 622151  000340
 829  001333' 151615  200000            MWORD   <CJV,J=1520,SOQ,OR,D=1,CENA,CCFZ>      ; 1516
 830  001334' 231400  020140
 831  001335' 151740  000000            MWORD   <JMAP,J=4000>                          ; 1517 error
 832  001336' 000000  000040
 833  001337' 152000  000000            MWORD   <CONT,A=16,B=3,SDQ,RMIN,D=3>           ; 1520
 834  001340' 623161  400340
 835  001341' 152115  230000            MWORD   <CJV,J=1523,SOQ,OR,D=1,CENA,CCFZ>      ; 1521
 836  001342' 231400  020140
 837  001343' 152240  000000            MWORD   <JMAP,J=4000>                          ; 1522 error
 838  001344' 000000  000040
 839  001345' 152300  000000            MWORD   <CONT,A=17,B=4,SDQ,RMIN,D=5>           ; 1523
 840  001346' 625172  000340
 841  001347' 152415  260000            MWORD   <CJV,J=1526,SOQ,OR,D=1,CENA,CCFZ>      ; 1524
 842  001350' 231400  020140
 843  001351' 152540  000000            MWORD   <JMAP,J=4000>                          ; 1525 error
 844  001352' 000000  000040
 845  001353' 152600  000000            MWORD   <CONT,A=0,B=5,SDQ,RMIN,D=7>            ; 1526
 846  001354' 627002  400340
 847  001355' 152715  310000            MWORD   <CJV,J=1531,SOQ,OR,D=1,CENA,CCFZ>      ; 1527
 848  001356' 231400  020140
 849  001357' 153040  000000            MWORD   <JMAP,J=4000>                          ; 1530 error
 850  001360' 000000  000040
 851  001361' 153100  000000            MWORD   <CONT,A=6,B=13,SDO,RMIN,D=1>           ; 1531
 852  001362' 721065  400340
 853  001363' 153215  340000            MWORD   <CJV,J=1534,SOQ,OR,D=1,CENA,CCFZ>      ; 1532
 854  001364' 231400  020140
 855  001365' 153340  000000            MWORD   <JMAP,J=4000>                          ; 1533 error
 856  001366' 000000  000040
 857  001367' 153400  000000            MWORD   <CONT,A=7,B=14,SDO,RMIN,D=2>           ; 1534
 858  001370' 722076  000340
 859  001371' 153515  370000            MWORD   <CJV,J=1537,SOQ,OR,D=1,CENA,CCFZ>      ; 1535
 860  001372' 231400  020140
 861  001373' 153640  000000            MWORD   <JMAP,J=4000>                          ; 1536 error
 862  001374' 000000  000040
 863  001375' 153700  000000            MWORD   <CONT,A=10,B=15,SDO,RMIN,D=3>          ; 1537
 864  001376' 723106  400340
 865  001377' 154015  420000            MWORD   <CJV,J=1542,SOQ,OR,D=1,CENA,CCFZ>      ; 1540
 866  001400' 231400  020140
 867  001401' 154140  000000            MWORD   <JMAP,J=4000>                          ; 1541 error
 868  001402' 000000  000040
 869  001403' 154200  000000            MWORD   <CONT,A=11,B=16,SDO,RMIN,D=5>          ; 1542
 870  001404' 725117  000340
 871  001405' 154315  450000            MWORD   <CJV,J=1545,SOQ,OR,D=1,CENA,CCFZ>      ; 1543
 872  001406' 231400  020140
 873  001407' 154440  000000            MWORD   <JMAP,J=4000>                          ; 1544 error
 874  001410' 000000  000040
 875  001411' 154500  000000            MWORD   <CONT,A=12,B=17,SDO,RMIN,D=7>          ; 1545
 876  001412' 727127  400340
 877  001413' 154615  500000            MWORD   <CJV,J=1550,SOQ,OR,D=1,CENA,CCFZ>      ; 1546
 878  001414' 231400  020140
```

```
879  001415' 154740  000000          MWORD   <JMAP,J=4000>                          ; 1547 error
880  001416' 000000  000040
881  001417' 155000  000000          MWORD   <CONT,A=0,B=5,SAQ,OR,D=1>              ; 1550
882  001420' 031002  400340
883  001421' 155115  530000          MWORD   <CJV,J=1553,SOQ,OR,D=1,CENA,CCFZ>      ; 1551
884  001422' 231400  020140
885  001423' 155240  000000          MWORD   <JMAP,J=4000>                          ; 1552 error
886  001424' 000000  000040
887  001425' 155300  000000          MWORD   <CONT,A=1,B=6,SAQ,OR,D=2>              ; 1553
888  001426' 032013  000340
889  001427' 155415  560000          MWORD   <CJV,J=1556,SOQ,OR,D=1,CENA,CCFZ>      ; 1554
890  001430' 231400  020140
891  001431' 155540  000000          MWORD   <JMAP,J=4000>                          ; 1555 error
892  001432' 000000  000040
893  001433' 155600  000000          MWORD   <CONT,A=2,B=7,SAQ,OR,D=3>              ; 1556
894  001434' 033023  400340
895  001435' 155715  610000          MWORD   <CJV,J=1561,SOQ,OR,D=1,CENA,CCFZ>      ; 1557
896  001436' 231400  020140
897  001437' 156040  000000          MWORD   <JMAP,J=4000>                          ; 1560 error
898  001440' 000000  000040
899  001441' 156100  000000          MWORD   <CONT,A=3,B=10,SAQ,OR,D=5>             ; 1561
900  001442' 035034  000340
901  001443' 156215  640000          MWORD   <CJV,J=1564,SOQ,OR,D=1,CENA,CCFZ>      ; 1562
902  001444' 231400  020140
903  001445' 156340  000000          MWORD   <JMAP,J=4000>                          ; 1563 error
904  001446' 000000  000040
905  001447' 156400  000000          MWORD   <CONT,A=4,B=11,SAQ,OR,D=7>             ; 1564
906  001450' 037044  400340
907  001451' 156515  670000          MWORD   <CJV,J=1567,SOQ,OR,D=1,CENA,CCFZ>      ; 1565
908  001452' 231400  020140
909  001453' 156640  000000          MWORD   <JMAP,J=4000>                          ; 1566 error
910  001454' 000000  000040
911  001455' 156700  000000          MWORD   <CONT,A=12,B=17,SAB,OR,D=1>            ; 1567
912  001456' 131127  400340
913  001457' 157015  720000          MWORD   <CJV,J=1572,SOQ,OR,D=1,CENA,CCFZ>      ; 1570
914  001460' 231400  020140
915  001461' 157140  000000          MWORD   <JMAP,J=4000>                          ; 1571 error
916  001462' 000000  000040
917  001463' 157200  000000          MWORD   <CONT,A=13,B=0,SAB,OR,D=2>             ; 1572
918  001464' 132130  000340
919  001465' 157315  750000          MWORD   <CJV,J=1575,SOQ,OR,D=1,CENA,CCFZ>      ; 1573
920  001466' 231400  020140
921  001467' 157440  000000          MWORD   <JMAP,J=4000>                          ; 1574 error
922  001470' 000000  000040
923  001471' 157500  000000          MWORD   <CONT,A=14,B=1,SAB,OR,D=3>             ; 1575
924  001472' 133140  400340
925  001473' 157616  000000          MWORD   <CJV,J=1600,SOQ,OR,D=1,CENA,CCFZ>      ; 1576
926  001474' 231400  020140
927  001475' 157740  000000          MWORD   <JMAP,J=4000>                          ; 1577 error
928  001476' 000000  000040
929  001477' 160000  000000          MWORD   <CONT,A=15,B=2,SAB,OR,D=5>             ; 1600
930  001500' 135151  000340
931  001501' 160116  030000          MWORD   <CJV,J=1603,SOQ,OR,D=1,CENA,CCFZ>      ; 1601
932  001502' 231400  020140
933  001503' 160240  000000          MWORD   <JMAP,J=4000>                          ; 1602 error
```

```
001504' 000000 000040
001505' 160300 000000              MWORD   <CONT,A=16,B=3,SAB,OR,D=7>              ; 1603
001506' 137161 400340
001507' 160416 060000              MWORD   <CJV,J=1606,SOQ,OR,D=1,CENA,CCFZ>       ; 1604
001510' 231400 020140
001511' 160540 000000              MWORD   <JMAP,J=4000>                           ; 1605 error
001512' 000000 000040
001513' 160600 000000              MWORD   <CONT,A=4,B=11,SOQ,OR,D=1>              ; 1606
001514' 231044 400340
001515' 160716 110000              MWORD   <CJV,J=1611,SOQ,OR,D=1,CENA,CCFZ>       ; 1607
001516' 231400 020140
001517' 161040 000000              MWORD   <JMAP,J=4000>                           ; 1610 error
001520' 000000 000040
001521' 161100 000000              MWORD   <CONT,A=5,B=12,SOQ,OR,D=2>              ; 1611
001522' 232055 000340
001523' 161216 140000              MWORD   <CJV,J=1614,SOQ,OR,D=1,CENA,CCFZ>       ; 1612
001524' 231400 020140
001525' 161340 000000              MWORD   <JMAP,J=4000>                           ; 1613 error
001526' 000000 000040
001527' 161400 000000              MWORD   <CONT,A=6,B=13,SOQ,OR,D=3>              ; 1614
001530' 233065 400340
001531' 161516 170000              MWORD   <CJV,J=1617,SOQ,OR,D=1,CENA,CCFZ>       ; 1615
001532' 231400 020140
001533' 161640 000000              MWORD   <JMAP,J=4000>                           ; 1616 error
001534' 000000 000040
001535' 161700 000000              MWORD   <CONT,A=7,B=14,SOQ,OR,D=5>              ; 1617
001536' 235076 000340
001537' 162016 220000              MWORD   <CJV,J=1622,SOQ,OR,D=1,CENA,CCFZ>       ; 1620
001540' 231400 020140
001541' 162140 000000              MWORD   <JMAP,J=4000>                           ; 1621 error
001542' 000000 000040
001543' 162200 000000              MWORD   <CONT,A=10,B=15,SOQ,OR,D=7>             ; 1622
001544' 237106 400340
001545' 162316 250000              MWORD   <CJV,J=1625,SOQ,OR,D=1,CENA,CCFZ>       ; 1623
001546' 231400 020140
001547' 162440 000000              MWORD   <JMAP,J=4000>                           ; 1624 error
001550' 000000 000040
001551' 162500 000000              MWORD   <CONT,A=16,B=3,SOB,OR,D=1>              ; 1625
001552' 331161 400340
001553' 162616 300000              MWORD   <CJV,J=1630,SOQ,OR,D=1,CENA,CCFZ>       ; 1626
001554' 231400 020140
001555' 162740 000000              MWORD   <JMAP,J=4000>                           ; 1627 error
001556' 000000 000040
001557' 163000 000000              MWORD   <CONT,A=17,B=4,SOB,OR,D=2>              ; 1630
001560' 332172 000340
001561' 163116 330000              MWORD   <CJV,J=1633,SOQ,OR,D=1,CENA,CCFZ>       ; 1631
001562' 231400 020140
001563' 163240 000000              MWORD   <JMAP,J=4000>                           ; 1632 error
001564' 000000 000040
001565' 163300 000000              MWORD   <CONT,A=0,B=5,SOB,OR,D=3>               ; 1633
001566' 333002 400340
001567' 163416 360000              MWORD   <CJV,J=1636,SOQ,OR,D=1,CENA,CCFZ>       ; 1634
001570' 231400 020140
001571' 163540 000000              MWORD   <JMAP,J=4000>                           ; 1635 error
001572' 000000 000040
```

```
 989  001573' 163600  000000            MWORD   <CONT,A=1,B=6,SOB,OR,D=5>                ; 1636
 990  001574' 335013  000340
 991  001575' 163716  410000            MWORD   <CJV,J=1641,SOQ,OR,D=1,CENA,CCFZ>        ; 1637
 992  001576' 231400  020140
 993  001577' 164040  000000            MWORD   <JMAP,J=4000>                            ; 1640 error
 994  001600' 000000  000040
 995  001601' 164100  000000            MWORD   <CONT,A=2,B=7,SOB,OR,D=7>                ; 1641
 996  001602' 337023  400340
 997  001603' 164216  440000            MWORD   <CJV,J=1644,SOQ,OR,D=1,CENA,CCFZ>        ; 1642
 998  001604' 231400  020140
 999  001605' 164340  000000            MWORD   <JMAP,J=4000>                            ; 1643 error
1000  001606' 000000  000040
1001  001607' 164400  000000            MWORD   <CONT,A=10,B=15,SOA,OR,D=1>              ; 1644
1002  001610' 431106  400340
1003  001611' 164516  470000            MWORD   <CJV,J=1647,SOQ,OR,D=1,CENA,CCFZ>        ; 1645
1004  001612' 231400  020140
1005  001613' 164640  000000            MWORD   <JMAP,J=4000>                            ; 1646 error
1006  001614' 000000  000040
1007  001615' 164700  000000            MWORD   <CONT,A=11,B=16,SOA,OR,D=2>              ; 1647
1008  001616' 432117  000340
1009  001617' 165016  520000            MWORD   <CJV,J=1652,SOQ,OR,D=1,CENA,CCFZ>        ; 1650
1010  001620' 231400  020140
1011  001621' 165140  000000            MWORD   <JMAP,J=4000>                            ; 1651 error
1012  001622' 000000  000040
1013  001623' 165200  000000            MWORD   <CONT,A=12,B=17,SOA,OR,D=3>              ; 1652
1014  001624' 433127  400340
1015  001625' 165316  550000            MWORD   <CJV,J=1655,SOQ,OR,D=1,CENA,CCFZ>        ; 1653
1016  001626' 231400  020140
1017  001627' 165440  000000            MWORD   <JMAP,J=4000>                            ; 1654 error
1018  001630' 000000  000040
1019  001631' 165500  000000            MWORD   <CONT,A=13,B=0,SOA,OR,D=5>               ; 1655
1020  001632' 435130  000340
1021  001633' 165616  600000            MWORD   <CJV,J=1660,SOQ,OR,D=1,CENA,CCFZ>        ; 1656
1022  001634' 231400  020140
1023  001635' 165740  000000            MWORD   <JMAP,J=4000>                            ; 1657 error
1024  001636' 000000  000040
1025  001637' 166000  000000            MWORD   <CONT,A=14,B=1,SOA,OR,D=7>               ; 1660
1026  001640' 437140  400340
1027  001641' 166116  630000            MWORD   <CJV,J=1663,SOQ,OR,D=1,CENA,CCFZ>        ; 1661
1028  001642' 231400  020140
1029  001643' 166240  000000            MWORD   <JMAP,J=4000>                            ; 1662 error
1030  001644' 000000  000040
1031  001645' 166300  000000            MWORD   <CONT,A=2,B=7,SDA,OR,D=1>                ; 1663
1032  001646' 531023  400340
1033  001647' 166416  660000            MWORD   <CJV,J=1666,SOQ,OR,D=1,CENA,CCFZ>        ; 1664
1034  001650' 231400  020140
1035  001651' 166540  000000            MWORD   <JMAP,J=4000>                            ; 1665 error
1036  001652' 000000  000040
1037  001653' 166600  000000            MWORD   <CONT,A=3,B=10,SDA,OR,D=2>               ; 1666
1038  001654' 532034  000340
1039  001655' 166716  710000            MWORD   <CJV,J=1671,SOQ,OR,D=1,CENA,CCFZ>        ; 1667
1040  001656' 231400  020140
1041  001657' 167040  000000            MWORD   <JMAP,J=4000>                            ; 1670 error
1042  001660' 000000  000040
1043  001661' 167100  000000            MWORD   <CONT,A=4,B=11,SDA,OR,D=3>               ; 1671
```

```
 1044  001662' 533044  400340
 1045  001663' 167216  740000             MWORD   <CJV,J=1674,SOQ,OR,D=1,CENA,CCFZ>     ; 1672
 1046  001664' 231400  020140
 1047  001665' 167340  000000             MWORD   <JMAP,J=4000>                         ; 1673 error
 1048  001666' 000000  000040
 1049  001667' 167400  000000             MWORD   <CONT,A=5,B=12,SDA,OR,D=5>            ; 1674
 1050  001670' 535055  000340
 1051  001671' 167516  770000             MWORD   <CJV,J=1677,SOQ,OR,D=1,CENA,CCFZ>     ; 1675
 1052  001672' 231400  020140
 1053  001673' 167640  000000             MWORD   <JMAP,J=4000>                         ; 1676 error
 1054  001674' 000000  000040
 1055  001675' 167700  000000             MWORD   <CONT,A=6,B=13,SDA,OR,D=7>            ; 1677
 1056  001676' 537065  400340
 1057  001677' 170017  020000             MWORD   <CJV,J=1702,SOQ,OR,D=1,CENA,CCFZ>     ; 1700
 1058  001700' 231400  020140
 1059  001701' 170140  000000             MWORD   <JMAP,J=4000>                         ; 1701 error
 1060  001702' 000000  000040
 1061  001703' 170200  000000             MWORD   <CONT,A=14,B=1,SDQ,OR,D=1>            ; 1702
 1062  001704' 631140  400340
 1063  001705' 170317  050000             MWORD   <CJV,J=1705,SOQ,OR,D=1,CENA,CCFZ>     ; 1703
 1064  001706' 231400  020140
 1065  001707' 170440  000000             MWORD   <JMAP,J=4000>                         ; 1704 error
 1066  001710' 000000  000040
 1067  001711' 170500  000000             MWORD   <CONT,A=15,B=2,SDQ,OR,D=2>            ; 1705
 1068  001712' 632151  000340
 1069  001713' 170617  100000             MWORD   <CJV,J=1710,SOQ,OR,D=1,CENA,CCFZ>     ; 1706
 1070  001714' 231400  020140
 1071  001715' 170740  000000             MWORD   <JMAP,J=4000>                         ; 1707 error
 1072  001716' 000000  000040
 1073  001717' 171000  000000             MWORD   <CONT,A=16,B=3,SDQ,OR,D=3>            ; 1710
 1074  001720' 634161  400340
 1075  001721' 171117  130000             MWORD   <CJV,J=1713,SOQ,OR,D=1,CENA,CCFZ>     ; 1711
 1076  001722' 231400  020140
 1077  001723' 171240  000000             MWORD   <JMAP,J=4000>                         ; 1712 error
 1078  001724' 000000  000040
 1079  001725' 171300  000000             MWORD   <CONT,A=17,B=4,SDQ,OR,D=5>            ; 1713
 1080  001726' 635172  000340
 1081  001727' 171417  160000             MWORD   <CJV,J=1716,SOQ,OR,D=1,CENA,CCFZ>     ; 1714
 1082  001730' 231400  020140
 1083  001731' 171500  000000             MWORD   <CONT,A=0,B=5,SDQ,OR,D=7>             ; 1715
 1084  001732' 637002  400340
 1085  001733' 171617  200000             MWORD   <CJV,J=1720,SOQ,OR,D=1,CENA,CCFZ>     ; 1716
 1086  001734' 231400  020140
 1087  001735' 171740  000000             MWORD   <JMAP,J=4000>                         ; 1717 error
 1088  001736' 000000  000040
 1089  001737' 172000  000000             MWORD   <CONT,A=6,B=13,SDO,OR,D=1>            ; 1720
 1090  001740' 731065  400340
 1091  001741' 172117  230000             MWORD   <CJV,J=1723,SOQ,OR,D=1,CENA,CCFZ>     ; 1721
 1092  001742' 231400  020140
 1093  001743' 172240  000000             MWORD   <JMAP,J=4000>                         ; 1722 error
 1094  001744' 000000  000040
 1095  001745' 172300  000000             MWORD   <CONT,A=7,B=14,SDO,OR,D=2>            ; 1723
 1096  001746' 732076  000340
 1097  001747' 172417  260000             MWORD   <CJV,J=1726,SOQ,OR,D=1,CENA,CCFZ>     ; 1724
 1098  001750' 231400  020140
```

```
001751' 172540  000000          MWORD   <JMAP,J=4000>                           ; 1725 error
001752' 000000  000040
001753' 172600  000000          MWORD   <CONT,A=10,B=15,SDO,OR,D=3>             ; 1726
001754' 733106  400340
001755' 172717  310000          MWORD   <CJV,J=1731,SOQ,OR,D=1,CENA,CCFZ>       ; 1727
001756' 231400  020140
001757' 173040  000000          MWORD   <JMAP,J=4000>                           ; 1730 error
001760' 000000  000040
001761' 173100  000000          MWORD   <CONT,A=11,B=16,SDO,OR,D=5>             ; 1731
001762' 735117  000340
001763' 173217  340000          MWORD   <CJV,J=1734,SOQ,OR,D=1,CENA,CCFZ>       ; 1732
001764' 231400  020140
001765' 173340  000000          MWORD   <JMAP,J=4000>                           ; 1733 error
001766' 000000  000040
001767' 173400  000000          MWORD   <CONT,A=12,B=17,SDO,OR,D=7>             ; 1734
001770' 737127  400340
001771' 173517  370000          MWORD   <CJV,J=1737,SOQ,OR,D=1,CENA,CCFZ>       ; 1735
001772' 231400  020140
001773' 173640  000000          MWORD   <JMAP,J=4000>                           ; 1736 error
001774' 000000  000040
001775' 173700  000000          MWORD   <CONT,A=0,B=5,SAQ,AND,D=1>              ; 1737
001776' 041002  400340
001777' 174017  420000          MWORD   <CJV,J=1742,SOQ,OR,D=1,CENA,CCFZ>       ; 1740
002000' 231400  020140
002001' 174140  000000          MWORD   <JMAP,J=4000>                           ; 1741 error
002002' 000000  000040
002003' 174200  000000          MWORD   <CONT,A=1,B=6,SAQ,AND,D=2>              ; 1742
002004' 042013  000340
002005' 174317  450000          MWORD   <CJV,J=1745,SOQ,OR,D=1,CENA,CCFZ>       ; 1743
002006' 231400  020140
002007' 174440  000000          MWORD   <JMAP,J=4000>                           ; 1744 error
002010' 000000  000040
002011' 174500  000000          MWORD   <CONT,A=2,B=7,SAQ,AND,D=3>              ; 1745
002012' 043023  400340
002013' 174617  500000          MWORD   <CJV,J=1750,SOQ,OR,D=1,CENA,CCFZ>       ; 1746
002014' 231400  020140
002015' 174740  000000          MWORD   <JMAP,J=4000>                           ; 1747 error
002016' 000000  000040
002017' 175000  000000          MWORD   <CONT,A=3,B=10,SAQ,AND,D=5>             ; 1750
002020' 045034  000340
002021' 175117  530000          MWORD   <CJV,J=1753,SOQ,OR,D=1,CENA,CCFZ>       ; 1751
002022' 231400  020140
002023' 175240  000000          MWORD   <JMAP,J=4000>                           ; 1752 error
002024' 000000  000040
002025' 175300  000000          MWORD   <CONT,A=4,B=11,SAQ,AND,D=7>             ; 1753
002026' 047044  400340
002027' 175417  560000          MWORD   <CJV,J=1756,SOQ,OR,D=1,CENA,CCFZ>       ; 1754
002030' 231400  020140
002031' 175540  000000          MWORD   <JMAP,J=4000>                           ; 1755 error
002032' 000000  000040
002033' 175600  000000          MWORD   <CONT,A=12,B=17,SAB,AND,D=1>            ; 1756
002034' 141127  400340
002035' 175717  610000          MWORD   <CJV,J=1761,SOQ,OR,D=1,CENA,CCFZ>       ; 1757
002036' 231400  020140
002037' 176040  000000          MWORD   <JMAP,J=4000>                           ; 1760 error
```

```
 1154  002040' 000000  000040
 1155  002041' 176100  000000          MWORD   <CONT,A=13,B=0,SAB,AND,D=2>             ; 1761
 1156  002042' 142130  000340
 1157  002043' 176217  640000          MWORD   <CJV,J=1764,SOQ,OR,D=1,CENA,CCFZ>       ; 1762
 1158  002044' 231400  020140
 1159  002045' 176340  000000          MWORD   <JMAP,J=4000>                           ; 1763 error
 1160  002046' 000000  000040
 1161  002047' 176400  000000          MWORD   <CONT,A=14,B=1,SAB,AND,D=3>             ; 1764
 1162  002050' 143140  400340
 1163  002051' 176517  670000          MWORD   <CJV,J=1767,SOQ,OR,D=1,CENA,CCFZ>       ; 1765
 1164  002052' 231400  020140
 1165  002053' 176640  000000          MWORD   <JMAP,J=4000>                           ; 1766 error
 1166  002054' 000000  000040
 1167  002055' 176700  000000          MWORD   <CONT,A=15,B=2,SAB,AND,D=5>             ; 1767
 1168  002056' 145151  000340
 1169  002057' 177017  720000          MWORD   <CJV,J=1772,SOQ,OR,D=1,CENA,CCFZ>       ; 1770
 1170  002060' 231400  020140
 1171  002061' 177140  000000          MWORD   <JMAP,J=4000>                           ; 1771 error
 1172  002062' 000000  000040
 1173  002063' 177200  000000          MWORD   <CONT,A=16,B=3,SAB,AND,D=7>             ; 1772
 1174  002064' 147161  400340
 1175  002065' 177317  750000          MWORD   <CJV,J=1775,SOQ,OR,D=1,CENA,CCFZ>       ; 1773
 1176  002066' 231400  020140
 1177  002067' 177440  000000          MWORD   <JMAP,J=4000>                           ; 1774 error
 1178  002070' 000000  000040
 1179  002071' 177500  000000          MWORD   <CONT,A=4,B=11,SOQ,AND,D=1>             ; 1775
 1180  002072' 241044  400340
 1181  002073' 177620  000000          MWORD   <CJV,J=2000,SOQ,OR,D=1,CENA,CCFZ>       ; 1776
 1182  002074' 231400  020140
 1183  002075' 177740  000000          MWORD   <JMAP,J=4000>                           ; 1777 error
 1184  002076' 000000  000040
 1185  002077' 200000  000000          MWORD   <CONT,A=5,B=12,SOQ,AND,D=2>             ; 2000
 1186  002100' 242055  000340
 1187  002101' 200120  030000          MWORD   <CJV,J=2003,SOQ,OR,D=1,CENA,CCFZ>       ; 2001
 1188  002102' 231400  020140
 1189  002103' 200240  000000          MWORD   <JMAP,J=4000>                           ; 2002 error
 1190  002104' 000000  000040
 1191  002105' 200300  000000          MWORD   <CONT,A=6,B=13,SOQ,AND,D=3>             ; 2003
 1192  002106' 243065  400340
 1193  002107' 200420  060000          MWORD   <CJV,J=2006,SOQ,OR,D=1,CENA,CCFZ>       ; 2004
 1194  002110' 231400  020140
 1195  002111' 200540  000000          MWORD   <JMAP,J=4000>                           ; 2005 error
 1196  002112' 000000  000040
 1197  002113' 200600  000000          MWORD   <CONT,A=7,B=14,SOQ,AND,D=5>             ; 2006
 1198  002114' 245076  000340
 1199  002115' 200720  110000          MWORD   <CJV,J=2011,SOQ,OR,D=1,CENA,CCFZ>       ; 2007
 1200  002116' 231400  020140
 1201  002117' 201040  000000          MWORD   <JMAP,J=4000>                           ; 2010 error
 1202  002120' 000000  000040
 1203  002121' 201100  000000          MWORD   <CONT,A=10,B=15,SOQ,AND,D=7>            ; 2011
 1204  002122' 247106  400340
 1205  002123' 201220  140000          MWORD   <CJV,J=2014,SOQ,OR,D=1,CENA,CCFZ>       ; 2012
 1206  002124' 231400  020140
 1207  002125' 201340  000000          MWORD   <JMAP,J=4000>                           ; 2013 error
 1208  002126' 000000  000040
```

```
1209 002127' 201400 000000          MWORD   <CONT,A=16,B=3,SOB,AND,D=1>             ; 2014
1210 002130' 341161 400340
1211 002131' 201520 170000          MWORD   <CJV,J=2017,SOQ,OR,D=1,CENA,CCFZ>       ; 2015
1212 002132' 231400 020140
1213 002133' 201640 000000          MWORD   <JMAP,J=4000>                           ; 2016 error
1214 002134' 000000 000040
1215 002135' 201700 000000          MWORD   <CONT,A=17,B=4,SOB,AND,D=2>             ; 2017
1216 002136' 342172 000340
1217 002137' 202020 220000          MWORD   <CJV,J=2022,SOQ,OR,D=1,CENA,CCFZ>       ; 2020
1218 002140' 231400 020140
1219 002141' 202140 000000          MWORD   <JMAP,J=4000>                           ; 2021 error
1220 002142' 000000 000040
1221 002143' 202200 000000          MWORD   <CONT,A=0,B=5,SOB,AND,D=3>              ; 2022
1222 002144' 343002 400340
1223 002145' 202320 250000          MWORD   <CJV,J=2025,SOQ,OR,D=1,CENA,CCFZ>       ; 2023
1224 002146' 231400 020140
1225 002147' 202440 000000          MWORD   <JMAP,J=4000>                           ; 2024 error
1226 002150' 000000 000040
1227 002151' 202500 000000          MWORD   <CONT,A=1,B=6,SOB,AND,D=5>              ; 2025
1228 002152' 345013 000340
1229 002153' 202620 300000          MWORD   <CJV,J=2030,SOQ,OR,D=1,CENA,CCFZ>       ; 2026
1230 002154' 231400 020140
1231 002155' 202740 000000          MWORD   <JMAP,J=4000>                           ; 2027 error
1232 002156' 000000 000040
1233 002157' 203000 000000          MWORD   <CONT,A=2,B=7,SOB,AND,D=7>              ; 2030
1234 002160' 347023 400340
1235 002161' 203120 330000          MWORD   <CJV,J=2033,SOQ,OR,D=1,CENA,CCFZ>       ; 2031
1236 002162' 231400 020140
1237 002163' 203240 000000          MWORD   <JMAP,J=4000>                           ; 2032 error
1238 002164' 000000 000040
1239 002165' 203300 000000          MWORD   <CONT,A=10,B=15,SOA,AND,D=1>            ; 2033
1240 002166' 441106 400340
1241 002167' 203420 360000          MWORD   <CJV,J=2036,SOQ,OR,D=1,CENA,CCFZ>       ; 2034
1242 002170' 231400 020140
1243 002171' 203540 000000          MWORD   <JMAP,J=4000>                           ; 2035 error
1244 002172' 000000 000040
1245 002173' 203600 000000          MWORD   <CONT,A=11,B=16,SOA,AND,D=2>            ; 2036
1246 002174' 442117 000340
1247 002175' 203720 410000          MWORD   <CJV,J=2041,SOQ,OR,D=1,CENA,CCFZ>       ; 2037
1248 002176' 231400 020140
1249 002177' 204040 000000          MWORD   <JMAP,J=4000>                           ; 2040 error
1250 002200' 000000 000040
1251 002201' 204100 000000          MWORD   <CONT,A=12,B=17,SOA,AND,D=3>            ; 2041
1252 002202' 443127 400340
1253 002203' 204220 440000          MWORD   <CJV,J=2044,SOQ,OR,D=1,CENA,CCFZ>       ; 2042
1254 002204' 231400 020140
1255 002205' 204340 000000          MWORD   <JMAP,J=4000>                           ; 2043 error
1256 002206' 000000 000040
1257 002207' 204400 000000          MWORD   <CONT,A=13,B=0,SOA,AND,D=5>             ; 2044
1258 002210' 445130 000340
1259 002211' 204520 470000          MWORD   <CJV,J=2047,SOQ,OR,D=1,CENA,CCFZ>       ; 2045
1260 002212' 231400 020140
1261 002213' 204640 000000          MWORD   <JMAP,J=4000>                           ; 2046 error
1262 002214' 000000 000040
1263 002215' 204700 000000          MWORD   <CONT,A=14,B=1,SOA,AND,D=7>             ; 2047
```

```
1264 002216' 447140 400340
1265 002217' 205020 520000              MWORD   <CJV,J=2052,SOQ,OR,D=1,CENA,CCFZ>  ; 2050
1266 002220' 231400 020140
1267 002221' 205140 000000              MWORD   <JMAP,J=4000>                      ; 2051 error
1268 002222' 000000 000040
1269 002223' 205200 000000              MWORD   <CONT,A=2,B=7,SDA,AND,D=1>         ; 2052
1270 002224' 541023 400340
1271 002225' 205320 550000              MWORD   <CJV,J=2055,SOQ,OR,D=1,CENA,CCFZ>  ; 2053
1272 002226' 231400 020140
1273 002227' 205440 000000              MWORD   <JMAP,J=4000>                      ; 2054 error
1274 002230' 000000 000040
1275 002231' 205500 000000              MWORD   <CONT,A=3,B=10,SDA,AND,D=2>        ; 2055
1276 002232' 542034 000340
1277 002233' 205620 600000              MWORD   <CJV,J=2060,SOQ,OR,D=1,CENA,CCFZ>  ; 2056
1278 002234' 231400 020140
1279 002235' 205740 000000              MWORD   <JMAP,J=4000>                      ; 2057 error
1280 002236' 000000 000040
1281 002237' 206000 000000              MWORD   <CONT,A=4,B=11,SDA,AND,D=3>        ; 2060
1282 002240' 543044 400340
1283 002241' 206120 630000              MWORD   <CJV,J=2063,SOQ,OR,D=1,CENA,CCFZ>  ; 2061
1284 002242' 231400 020140
1285 002243' 206240 000000              MWORD   <JMAP,J=4000>                      ; 2062 error
1286 002244' 000000 000040
1287 002245' 206300 000000              MWORD   <CONT,A=5,B=12,SDA,AND,D=5>        ; 2063
1288 002246' 545055 000340
1289 002247' 206420 660000              MWORD   <CJV,J=2066,SOQ,OR,D=1,CENA,CCFZ>  ; 2064
1290 002250' 231400 020140
1291 002251' 206540 000000              MWORD   <JMAP,J=4000>                      ; 2065 error
1292 002252' 000000 000040
1293 002253' 206600 000000              MWORD   <CONT,A=6,B=13,SDA,AND,D=7>        ; 2066
1294 002254' 547065 400340
1295 002255' 206720 710000              MWORD   <CJV,J=2071,SOQ,OR,D=1,CENA,CCFZ>  ; 2067
1296 002256' 231400 020140
1297 002257' 207040 000000              MWORD   <JMAP,J=4000>                      ; 2070 error
1298 002260' 000000 000040
1299 002261' 207100 000000              MWORD   <CONT,A=14,B=1,SDQ,AND,D=1>        ; 2071
1300 002262' 641140 400340
1301 002263' 207220 740000              MWORD   <CJV,J=2074,SOQ,OR,D=1,CENA,CCFZ>  ; 2072
1302 002264' 231400 020140
1303 002265' 207340 000000              MWORD   <JMAP,J=4000>                      ; 2073 error
1304 002266' 000000 000040
1305 002267' 207400 000000              MWORD   <CONT,A=15,B=2,SDQ,AND,D=2>        ; 2074
1306 002270' 642151 000340
1307 002271' 207520 770000              MWORD   <CJV,J=2077,SOQ,OR,D=1,CENA,CCFZ>  ; 2075
1308 002272' 231400 020140
1309 002273' 207640 000000              MWORD   <JMAP,J=4000>                      ; 2076 error
1310 002274' 000000 000040
1311 002275' 207700 000000              MWORD   <CONT,A=16,B=3,SDQ,AND,D=3>        ; 2077
1312 002276' 643161 400340
1313 002277' 210021 020000              MWORD   <CJV,J=2102,SOQ,OR,D=1,CENA,CCFZ>  ; 2100
1314 002300' 231400 020140
1315 002301' 210140 000000              MWORD   <JMAP,J=4000>                      ; 2101 error
1316 002302' 000000 000040
1317 002303' 210200 000000              MWORD   <CONT,A=17,B=4,SDQ,AND,D=5>        ; 2102
1318 002304' 645172 000340
```

```
1319 002305' 210321 050000             MWORD   <CJV,J=2105,SOQ,OR,D=1,CENA,CCFZ>    ; 2103
1320 002306' 231400 020140
1321 002307' 210440 000000             MWORD   <JMAP,J=4000>                        ; 2104 error
1322 002310' 000000 000040
1323 002311' 210500 000000             MWORD   <CONT,A=0,B=5,SDQ,AND,D=7>           ; 2105
1324 002312' 647002 400340
1325 002313' 210621 100000             MWORD   <CJV,J=2110,SOQ,OR,D=1,CENA,CCFZ>    ; 2106
1326 002314' 231400 020140
1327 002315' 210740 000000             MWORD   <JMAP,J=4000>                        ; 2107 error
1328 002316' 000000 000040
1329 002317' 211000 000000             MWORD   <CONT,A=6,B=13,SDO,AND,D=1>          ; 2110
1330 002320' 741065 400340
1331 002321' 211121 130000             MWORD   <CJV,J=2113,SOQ.OR,D=1,CENA,CCFZ>    ; 2111
1332 002322' 231400 020140
1333 002323' 211240 000000             MWORD   <JMAP,J=4000>                        ; 2112 error
1334 002324' 000000 000040
1335 002325' 211300 000000             MWORD   <CONT,A=7,B=14,SDO,AND,D=2>          ; 2113
1336 002326' 742076 000340
1337 002327' 211421 160000             MWORD   <CJV,J=2116,SOQ,OR,D=1,CENA,CCFZ>    ; 2114
1338 002330' 231400 020140
1339 002331' 211540 000000             MWORD   <JMAP,J=4000>                        ; 2115 error
1340 002332' 000000 000040
1341 002333' 211600 000000             MWORD   <CONT,A=10,B=15,SDO,AND,D=3>         ; 2116
1342 002334' 743106 400340
1343 002335' 211721 210000             MWORD   <CJV,J=2121,SOQ,OR,D=1,CENA,CCFZ>    ; 2117
1344 002336' 231400 020140
1345 002337' 212040 000000             MWORD   <JMAP,J=4000>                        ; 2120 error
1346 002340' 000000 000040
1347 002341' 212100 000000             MWORD   <CONT,A=11,B=16,SDO,AND,D=5>         ; 2121
1348 002342' 745117 000340
1349 002343' 212221 240000             MWORD   <CJV,J=2124,SOQ,OR,D=1,CENA,CCFZ>    ; 2122
1350 002344' 231400 020140
1351 002345' 212340 000000             MWORD   <JMAP,J=4000>                        ; 2123 error
1352 002346' 000000 000040
1353 002347' 212400 000000             MWORD   <CONT,A=12,B=17,SDO,AND,D=7>         ; 2124
1354 002350' 747127 400340
1355 002351' 212521 270000             MWORD   <CJV,J=2127,SOQ,OR,D=1,CENA,CCFZ>    ; 2125
1356 002352' 231400 020140
1357 002353' 212640 000000             MWORD   <JMAP,J=4000>                        ; 2126 error
1358 002354' 000000 000040
1359 002355' 212700 000000             MWORD   <CONT,A=0,B=5,SAQ,NAND,D=1>          ; 2127
1360 002356' 051002 400340
1361 002357' 213021 320000             MWORD   <CJV,J=2132,SOQ,OR,D=1,CENA,CCFZ>    ; 2130
1362 002360' 231400 020140
1363 002361' 213140 000000             MWORD   <JMAP,J=4000>                        ; 2131 error
1364 002362' 000000 000040
1365 002363' 213200 000000             MWORD   <CONT,A=1,B=6,SAQ,NAND,D=2>          ; 2132
1366 002364' 052013 000340
1367 002365' 213321 350000             MWORD   <CJV,J=2135,SOQ,OR,D=1,CENA,CCFZ>    ; 2133
1368 002366' 231400 020140
1369 002367' 213440 000000             MWORD   <JMAP,J=4000>                        ; 2134 error
1370 002370' 000000 000040
1371 002371' 213500 000000             MWORD   <CONT,A=2,B=7,SAQ,NAND,D=3>          ; 2135
1372 002372' 053023 400340
1373 002373' 213621 400000             MWORD   <CJV,J=2140,SOQ,OR,D=1,CENA,CCFZ>    ; 2136
```

```
002374' 231400  020140
002375' 213740  000000          MWORD   <JMAP,J=4000>                           ; 2137 error
002376' 000000  000040
002377' 214000  000000          MWORD   <CONT,A=3,B=10,SAQ,NAND,D=5>            ; 2140
002400' 055034  000340
002401' 214121  430000          MWORD   <CJV,J=2143,SOQ,OR,D=1,CENA,CCFZ>       ; 2141
002402' 231400  020140
002403' 214240  000000          MWORD   <JMAP,J=4000>                           ; 2142 error
002404' 000000  000040
002405' 214300  000000          MWORD   <CONT,A=4,B=11,SAQ,NAND,D=7>            ; 2143
002406' 057044  400340
002407' 214421  460000          MWORD   <CJV,J=2146,SOQ,OR,D=1,CENA,CCFZ>       ; 2144
002410' 231400  020140
002411' 214540  000000          MWORD   <JMAP,J=4000>                           ; 2145 error
002412' 000000  000040
002413' 214600  000000          MWORD   <CONT,A=12,B=17,SAB,NAND,D=1>           ; 2146
002414' 151127  400340
002415' 214721  510000          MWORD   <CJV,J=2151,SOQ,OR,D=1,CENA,CCFZ>       ; 2147
002416' 231400  020140
002417' 215040  000000          MWORD   <JMAP,J=4000>                           ; 2150 error
002420' 000000  000040
002421' 215100  000000          MWORD   <CONT,A=13,B=0,SAB,NAND,D=2>            ; 2151
002422' 152130  000340
002423' 215221  540000          MWORD   <CJV,J=2154,SOQ,OR,D=1,CENA,CCFZ>       ; 2152
002424' 231400  020140
002425' 215340  000000          MWORD   <JMAP,J=4000>                           ; 2153 error
002426' 000000  000040
002427' 215400  000000          MWORD   <CONT,A=14,B=1,SAB,NAND,D=3>            ; 2154
002430' 153140  400340
002431' 215521  570000          MWORD   <CJV,J=2157,SOQ,OR,D=1,CENA,CCFZ>       ; 2155
002432' 231400  020140
002433' 215640  000000          MWORD   <JMAP,J=4000>                           ; 2156 error
002434' 000000  000040
002435' 215700  000000          MWORD   <CONT,A=15,B=2,SAB,NAND,D=5>            ; 2157
002436' 155151  000340
002437' 216021  620000          MWORD   <CJV,J=2162,SOQ,OR,D=1,CENA,CCFZ>       ; 2160
002440' 231400  020140
002441' 216140  000000          MWORD   <JMAP,J=4000>                           ; 2161 error
002442' 000000  000040
002443' 216200  000000          MWORD   <CONT,A=16,B=3,SAB,NAND,D=7>            ; 2162
002444' 157161  400340
002445' 216321  650000          MWORD   <CJV,J=2165,SOQ,OR,D=1,CENA,CCFZ>       ; 2163
002446' 231400  020140
002447' 216440  000000          MWORD   <JMAP,J=4000>                           ; 2164 error
002450' 000000  000040
002451' 216500  000000          MWORD   <CONT,A=4,B=11,SOQ,NAND,D=1>            ; 2165
002452' 251044  400340
002453' 216621  700000          MWORD   <CJV,J=2170,SOQ,OR,D=1,CENA,CCFZ>       ; 2166
002454' 231400  020140
002455' 216740  000000          MWORD   <JMAP,J=4000>                           ; 2167 error
002456' 000000  000040
002457' 217000  000000          MWORD   <CONT,A=5,B=12,SOQ,NAND,D=2>            ; 2170
002460' 252055  000340
002461' 217121  730000          MWORD   <CJV,J=2173,SOQ,OR,D=1,CENA,CCFZ>       ; 2171
002462' 231400  020140
```

```
002463' 217240  000000     MWORD   <JMAP,J=4000>                              ; 2172 error
002464' 000000  000040
002465' 217300  000000     MWORD   <CONT,A=6,B=13,SOQ,NAND,D=3>               ; 2173
002466' 253065  400340
002467' 217421  760000     MWORD   <CJV,J=2176,SOQ,OR,D=1,CENA,CCFZ>          ; 2174
002470' 231400  020140
002471' 217540  000000     MWORD   <JMAP,J=4000>                              ; 2175 error
002472' 000000  000040
002473' 217600  000000     MWORD   <CONT,A=7,B=14,SOQ,NAND,D=5>               ; 2176
002474' 255076  000340
002475' 217722  010000     MWORD   <CJV,J=2201,SOQ,OR,D=1,CENA,CCFZ>          ; 2177
002476' 231400  020140
002477' 220040  000000     MWORD   <JMAP,J=4000>                              ; 2200 error
002500' 000000  000040
002501' 220100  000000     MWORD   <CONT,A=10,B=15,SOQ,NAND,D=7>              ; 2201
002502' 257106  400340
002503' 220222  040000     MWORD   <CJV,J=2204,SOQ,OR,D=1,CENA,CCFZ>          ; 2202
002504' 231400  020140
002505' 220340  000000     MWORD   <JMAP,J=4000>                              ; 2203 error
002506' 000000  000040
002507' 220400  000000     MWORD   <CONT,A=16,B=3,SOB,NAND,D=1>               ; 2204
002510' 351161  400340
002511' 220522  070000     MWORD   <CJV,J=2207,SOQ,OR,D=1,CENA,CCFZ>          ; 2205
002512' 231400  020140
002513' 220640  000000     MWORD   <JMAP,J=4000>                              ; 2206 error
002514' 000000  000040
002515' 220700  000000     MWORD   <CONT,A=17,B=4,SOB,NAND,D=2>               ; 2207
002516' 352172  000340
002517' 221022  120000     MWORD   <CJV,J=2212,SOQ,OR,D=1,CENA,CCFZ>          ; 2210
002520' 231400  020140
002521' 221140  000000     MWORD   <JMAP,J=4000>                              ; 2211 error
002522' 000000  000040
002523' 221200  000000     MWORD   <CONT,A=0,B=5,SOB,NAND,D=3>                ; 2212
002524' 353002  400340
002525' 221322  150000     MWORD   <CJV,J=2215,SOQ,OR,D=1,CENA,CCFZ>          ; 2213
002526' 231400  020140
002527' 221440  000000     MWORD   <JMAP,J=4000>                              ; 2214 error
002530' 000000  000040
002531' 221500  000000     MWORD   <CONT,A=1,B=6,SOB,NAND,D=5>                ; 2215
002532' 355013  000340
002533' 221622  200000     MWORD   <CJV,J=2220,SOQ,OR,D=1,CENA,CCFZ>          ; 2216
002534' 231400  020140
002535' 221740  000000     MWORD   <JMAP,J=4000>                              ; 2217 error
002536' 000000  000040
002537' 222000  000000     MWORD   <CONT,A=2,B=7,SOB,NAND,D=7>                ; 2220
002540' 357023  400340
002541' 222122  230000     MWORD   <CJV,J=2223,SOQ,OR,D=1,CENA,CCFZ>          ; 2221
002542' 231400  020140
002543' 222240  000000     MWORD   <JMAP,J=4000>                              ; 2222 error
002544' 000000  000040
002545' 222300  000000     MWORD   <CONT,A=10,B=15,SOA,NAND,D=1>              ; 2223
002546' 451106  400340
002547' 222422  260000     MWORD   <CJV,J=2226,SOQ,OR,D=1,CENA,CCFZ>          ; 2224
002550' 231400  020140
002551' 222540  000000     MWORD   <JMAP,J=4000>                              ; 2225 error
```

```
DFPTA
DFPTA'

AAPNT
AEXEC
ALU
CALL
E23
EBUS
ERFLG
GO
IPACL
LAST
MA31
MLAST
NDMP
RTN
SCOPE
T31M
TA31
TG31
TLOAD
TRACE
TSTA3
TSTA3
TSTA3
TSTA3
TSTA4
TSTA4
TSTSU
TX31
TXALL
Z5
ZALU
$ARG2
$B
$CHR
$GARG
%ADDR
%ML
%MR
.LA
.LADD
.LAND
.LB
.LBAD
.LCCF
.LCEN
.LCJV
.LCON
.LD
.LJ
.LJMA
.LMG(
.LNAN
.LOR
.LPLU
.LRM.
```

```
002552' 000000  000040
002553' 222600  000000                  MWORD   <CONT,A=11,B=16,SOA,NAND,D=2>            ; 2226
002554' 452117  000340
002555' 222722  310000                  MWORD   <CJV,J=2231,SOQ,OR,D=1,CENA,CCFZ>        ; 2227
002556' 231400  020140
002557' 223040  000000                  MWORD   <JMAP,J=4000>                            ; 2230 error
002560' 000000  000040
002561' 223100  000000                  MWORD   <CONT,A=12,B=17,SOA,NAND,D=3>            ; 2231
002562' 453127  400340
002563' 223222  340000                  MWORD   <CJV,J=2234,SOQ,OR,D=1,CENA,CCFZ>        ; 2232
002564' 231400  020140
002565' 223340  000000                  MWORD   <JMAP,J=4000>                            ; 2233 error
002566' 000000  000040
002567' 223400  000000                  MWORD   <CONT,A=13,B=0,SOA,NAND,D=5>             ; 2234
002570' 455130  000340
002571' 223522  370000                  MWORD   <CJV,J=2237,SOQ,OR,D=1,CENA,CCFZ>        ; 2235
002572' 231400  020140
002573' 223640  000000                  MWORD   <JMAP,J=4000>                            ; 2236 error
002574' 000000  000040
002575' 223700  000000                  MWORD   <CONT,A=14,B=1,SOA,NAND,D=7>             ; 2237
002576' 457140  400340
002577' 224022  420000                  MWORD   <CJV,J=2242,SOQ,OR,D=1,CENA,CCFZ>        ; 2240
002600' 231400  020140
002601' 224140  000000                  MWORD   <JMAP,J=4000>                            ; 2241 error
002602' 000000  000040
002603' 224200  000000                  MWORD   <CONT,A=2,B=7,SDA,NAND,D=1>              ; 2242
002604' 551023  400340
002605' 224322  450000                  MWORD   <CJV,J=2245,SOQ,OR,D=1,CENA,CCFZ>        ; 2243
002606' 231400  020140
002607' 224440  000000                  MWORD   <JMAP,J=4000>                            ; 2244 error
002610' 000000  000040
002611' 224500  000000                  MWORD   <CONT,A=3,B=10,SDA,NAND,D=2>             ; 2245
002612' 552034  000340
002613' 224622  500000                  MWORD   <CJV,J=2250,SOQ,OR,D=1,CENA,CCFZ>        ; 2246
002614' 231400  020140
002615' 224740  000000                  MWORD   <JMAP,J=4000>                            ; 2247 error
002616' 000000  000040
002617' 225000  000000                  MWORD   <CONT,A=4,B=11,SDA,NAND,D=3>             ; 2250
002620' 553044  400340
002621' 225122  530000                  MWORD   <CJV,J=2253,SOQ,OR,D=1,CENA,CCFZ>        ; 2251
002622' 231400  020140
002623' 225240  000000                  MWORD   <JMAP,J=4000>                            ; 2252 error
002624' 000000  000040
002625' 225300  000000                  MWORD   <CONT,A=5,B=12,SDA,NAND,D=5>             ; 2253
002626' 555055  000340
002627' 225422  560000                  MWORD   <CJV,J=2256,SOQ,OR,D=1,CENA,CCFZ>        ; 2254
002630' 231400  020140
002631' 225540  000000                  MWORD   <JMAP,J=4000>                            ; 2255 error
002632' 000000  000040
002633' 225600  000000                  MWORD   <CONT,A=6,B=13,SDA,NAND,D=7>             ; 2256
002634' 557065  400340
002635' 225722  610000                  MWORD   <CJV,J=2261,SOQ,OR,D=1,CENA,CCFZ>        ; 2257
002636' 231400  020140
002637' 226040  000000                  MWORD   <JMAP,J=4000>                            ; 2260 error
002640' 000000  000040
```

```
 1539  002641' 226100  000000          MWORD   <CONT,A=14,B=1,SDQ,NAND,D=1>          ; 2261
 1540  002642' 651140  400340
 1541  002643' 226222  640000          MWORD   <CJV,J=2264,SOQ,OR,D=1,CENA,CCFZ>     ; 2262
 1542  002644' 231400  020140
 1543  002645' 226340  000000          MWORD   <JMAP,J=4000>                         ; 2263 error
 1544  002646' 000000  000040
 1545  002647' 226400  000000          MWORD   <CONT,A=15,B=2,SDQ,NAND,D=2>          ; 2264
 1546  002650' 652151  000340
 1547  002651' 226522  670000          MWORD   <CJV,J=2267,SOQ,OR,D=1,CENA,CCFZ>     ; 2265
 1548  002652' 231400  020140
 1549  002653' 226640  000000          MWORD   <JMAP,J=4000>                         ; 2266 error
 1550  002654' 000000  000040
 1551  002655' 226700  000000          MWORD   <CONT,A=16,B=3,SDQ,NAND,D=3>          ; 2267
 1552  002656' 653161  400340
 1553  002657' 227022  720000          MWORD   <CJV,J=2272,SOQ,OR,D=1,CENA,CCFZ>     ; 2270
 1554  002660' 231400  020140
 1555  002661' 227140  000000          MWORD   <JMAP,J=4000>                         ; 2271 error
 1556  002662' 000000  000040
 1557  002663' 227200  000000          MWORD   <CONT,A=17,B=4,SDQ,NAND,D=5>          ; 2272
 1558  002664' 655172  000340
 1559  002665' 227322  750000          MWORD   <CJV,J=2275,SOQ,OR,D=1,CENA,CCFZ>     ; 2273
 1560  002666' 231400  020140
 1561  002667' 227440  000000          MWORD   <JMAP,J=4000>                         ; 2274 error
 1562  002670' 000000  000040
 1563  002671' 227500  000000          MWORD   <CONT,A=0,B=5,SDQ,NAND,D=7>           ; 2275
 1564  002672' 657002  400340
 1565  002673' 227623  000000          MWORD   <CJV,J=2300,SOQ,OR,D=1,CENA,CCFZ>     ; 2276
 1566  002674' 231400  020140
 1567  002675' 227740  000000          MWORD   <JMAP,J=4000>                         ; 2277 error
 1568  002676' 000000  000040
 1569  002677' 230000  000000          MWORD   <CONT,A=6,B=13,SDO,NAND,D=1>          ; 2300
 1570  002700' 751065  400340
 1571  002701' 230123  030000          MWORD   <CJV,J=2303,SOQ,OR,D=1,CENA,CCFZ>     ; 2301
 1572  002702' 231400  020140
 1573  002703' 230240  000000          MWORD   <JMAP,J=4000>                         ; 2302 error
 1574  002704' 000000  000040
 1575  002705' 230300  000000          MWORD   <CONT,A=7,B=14,SDO,NAND,D=2>          ; 2303
 1576  002706' 752076  000340
 1577  002707' 230423  060000          MWORD   <CJV,J=2306,SOQ,OR,D=1,CENA,CCFZ>     ; 2304
 1578  002710' 231400  020140
 1579  002711' 230540  000000          MWORD   <JMAP,J=4000>                         ; 2305 error
 1580  002712' 000000  000040
 1581  002713' 230600  000000          MWORD   <CONT,A=10,B=15,SDO,NAND,D=3>         ; 2306
 1582  002714' 753106  400340
 1583  002715' 230723  110000          MWORD   <CJV,J=2311,SOQ,OR,D=1,CENA,CCFZ>     ; 2307
 1584  002716' 231400  020140
 1585  002717' 231040  000000          MWORD   <JMAP,J=4000>                         ; 2310 error
 1586  002720' 000000  000040
 1587  002721' 231100  000000          MWORD   <CONT,A=11,B=16,SDO,NAND,D=5>         ; 2311
 1588  002722' 755117  000340
 1589  002723' 231223  140000          MWORD   <CJV,J=2314,SOQ,OR,D=1,CENA,CCFZ>     ; 2312
 1590  002724' 231400  020140
 1591  002725' 231340  000000          MWORD   <JMAP,J=4000>                         ; 2313 error
 1592  002726' 000000  000040
 1593  002727' 231400  000000          MWORD   <CONT,A=12,B=17,SDO,NAND,D=7>         ; 2314
```

```
002730' 757127  400340
002731' 231523  170000                          MWORD   <CJV,J=2317,SOQ,OR,D=1,CENA,CCFZ>     ; 2315
002732' 231400  020140
002733' 231640  000000                          MWORD   <JMAP,J=4000>                         ; 2316 error
002734' 000000  000040
002735' 231700  000000                          MWORD   <CONT,A=0,B=5,SAQ,XOR,D=1>            ; 2317
002736' 061002  400340
002737' 232023  220000                          MWORD   <CJV,J=2322,SOQ,OR,D=1,CENA,CCFZ>     ; 2320
002740' 231400  020140
002741' 232140  000000                          MWORD   <JMAP,J=4000>                         ; 2321 error
002742' 000000  000040
002743' 232200  000000                          MWORD   <CONT,A=1,B=6,SAQ,XOR,D=2>            ; 2322
002744' 062013  000340
002745' 232323  250000                          MWORD   <CJV,J=2325,SOQ,OR,D=1,CENA,CCFZ>     ; 2323
002746' 231400  020140
002747' 232440  000000                          MWORD   <JMAP,J=4000>                         ; 2324 error
002750' 000000  000040
002751' 232500  000000                          MWORD   <CONT,A=2,B=7,SAQ,XOR,D=3>            ; 2325
002752' 063023  400340
002753' 232623  300000                          MWORD   <CJV,J=2330,SOQ,OR,D=1,CENA,CCFZ>     ; 2326
002754' 231400  020140
002755' 232740  000000                          MWORD   <JMAP,J=4000>                         ; 2327 error
002756' 000000  000040
002757' 233000  000000                          MWORD   <CONT,A=3,B=10,SAQ,XOR,D=5>           ; 2330
002760' 065034  000340
002761' 233123  330000                          MWORD   <CJV,J=2333,SOQ,OR,D=1,CENA,CCFZ>     ; 2331
002762' 231400  020140
002763' 233240  000000                          MWORD   <JMAP,J=4000>                         ; 2332 error
002764' 000000  000040
002765' 233300  000000                          MWORD   <CONT,A=4,B=11,SAQ,XOR,D=7>           ; 2333
002766' 067044  400340
002767' 233423  360000                          MWORD   <CJV,J=2336,SOQ,OR,D=1,CENA,CCFZ>     ; 2334
002770' 231400  020140
002771' 233540  000000                          MWORD   <JMAP,J=4000>                         ; 2335 error
002772' 000000  000040
002773' 233600  000000                          MWORD   <CONT,A=12,B=17,SAB,XOR,D=1>          ; 2336
002774' 161127  400340
002775' 233723  410000                          MWORD   <CJV,J=2341,SOQ,OR,D=1,CENA,CCFZ>     ; 2337
002776' 231400  020140
002777' 234040  000000                          MWORD   <JMAP,J=4000>                         ; 2340 error
003000' 000000  000040
003001' 234100  000000                          MWORD   <CONT,A=13,B=0,SAB,XOR,D=2>           ; 2341
003002' 162130  000340
003003' 234223  440000                          MWORD   <CJV,J=2344,SOQ,OR,D=1,CENA,CCFZ>     ; 2342
003004' 231400  020140
003005' 234340  000000                          MWORD   <JMAP,J=4000>                         ; 2343 error
003006' 000000  000040
003007' 234400  000000                          MWORD   <CONT,A=14,B=1,SAB,XOR,D=3>           ; 2344
003010' 163140  400340
003011' 234523  470000                          MWORD   <CJV,J=2347,SOQ,OR,D=1,CENA,CCFZ>     ; 2345
003012' 231400  020140
003013' 234640  000000                          MWORD   <JMAP,J=4000>                         ; 2346 error
003014' 000000  000040
003015' 234700  000000                          MWORD   <CONT,A=15,B=2,SAB,XOR,D=5>           ; 2347
003016' 165151  000340
```

```
003017' 235023  520000          MWORD   <CJV,J=2352,SOQ,OR,D=1,CENA,CCFZ>   ; 2350
003020' 231400  020140
003021' 235140  000000          MWORD   <JMAP,J=4000>                       ; 2351 error
003022' 000000  000040
003023' 235200  000000          MWORD   <CONT,A=16,B=3,SAB,XOR,D=7>         ; 2352
003024' 167161  400340
003025' 235323  550000          MWORD   <CJV,J=2355,SOQ,OR,D=1,CENA,CCFZ>   ; 2353
003026' 231400  020140
003027' 235440  000000          MWORD   <JMAP,J=4000>                       ; 2354 error
003030' 000000  000040
003031' 235500  000000          MWORD   <CONT,A=4,B=11,SOQ,XOR,D=1>         ; 2355
003032' 261044  400340
003033' 235623  600000          MWORD   <CJV,J=2360,SOQ,OR,D=1,CENA,CCFZ>   ; 2356
003034' 231400  020140
003035' 235740  000000          MWORD   <JMAP,J=4000>                       ; 2357 error
003036' 000000  000040
003037' 236000  000000          MWORD   <CONT,A=5,B=12,SOQ,XOR,D=2>         ; 2360
003040' 262055  000340
003041' 236123  630000          MWORD   <CJV,J=2363,SOQ,OR,D=1,CENA,CCFZ>   ; 2361
003042' 231400  020140
003043' 236240  000000          MWORD   <JMAP,J=4000>                       ; 2362 error
003044' 000000  000040
003045' 236300  000000          MWORD   <CONT,A=6,B=13,SOQ,XOR,D=3>         ; 2363
003046' 263065  400340
003047' 236423  660000          MWORD   <CJV,J=2366,SOQ,OR,D=1,CENA,CCFZ>   ; 2364
003050' 231400  020140
003051' 236540  000000          MWORD   <JMAP,J=4000>                       ; 2365 error
003052' 000000  000040
003053' 236600  000000          MWORD   <CONT,A=7,B=14,SOQ,XOR,D=5>         ; 2366
003054' 265076  000340
003055' 236723  710000          MWORD   <CJV,J=2371,SOQ,OR,D=1,CENA,CCFZ>   ; 2367
003056' 231400  020140
003057' 237040  000000          MWORD   <JMAP,J=4000>                       ; 2370 error
003060' 000000  000040
003061' 237100  000000          MWORD   <CONT,A=10,B=15,SOQ,XOR,D=7>        ; 2371
003062' 267106  400340
003063' 237223  740000          MWORD   <CJV,J=2374,SOQ,OR,D=1,CENA,CCFZ>   ; 2372
003064' 231400  020140
003065' 237340  000000          MWORD   <JMAP,J=4000>                       ; 2373 error
003066' 000000  000040
003067' 237400  000000          MWORD   <CONT,A=16,B=3,SOB,XOR,D=1>         ; 2374
003070' 361161  400340
003071' 237523  770000          MWORD   <CJV,J=2377,SOQ,OR,D=1,CENA,CCFZ>   ; 2375
003072' 231400  020140
003073' 237640  000000          MWORD   <JMAP,J=4000>                       ; 2376 error
003074' 000000  000040
003075' 237700  000000          MWORD   <CONT,A=17,B=4,SOB,XOR,D=2>         ; 2377
003076' 362172  000340
003077' 240024  020000          MWORD   <CJV,J=2402,SOQ,OR,D=1,CENA,CCFZ>   ; 2400
003100' 231400  020140
003101' 240140  000000          MWORD   <JMAP,J=4000>                       ; 2401 error
003102' 000000  000040
003103' 240200  000000          MWORD   <CONT,A=0,B=5,SOB,XOR,D=3>          ; 2402
003104' 363002  400340
003105' 240324  050000          MWORD   <CJV,J=2405,SOQ,OR,D=1,CENA,CCFZ>   ; 2403
```

```
 1704  003106' 231400  020140
 1705  003107' 240440  000000                   MWORD   <JMAP,J=4000>                           ; 2404 error
 1706  003110' 000000  000040
 1707  003111' 240500  000000                   MWORD   <CONT,A=1,B=6,SOB,XOR,D=5>              ; 2405
 1708  003112' 365013  000340
 1709  003113' 240624  100000                   MWORD   <CJV,J=2410,SOQ,OR,D=1,CENA,CCFZ>       ; 2406
 1710  003114' 231400  020140
 1711  003115' 240740  000000                   MWORD   <JMAP,J=4000>                           ; 2407 error
 1712  003116' 000000  000040
 1713  003117' 241000  000000                   MWORD   <CONT,A=2,B=7,SOB,XOR,D=7>              ; 2410
 1714  003120' 367023  400340
 1715  003121' 241124  130000                   MWORD   <CJV,J=2413,SOQ,OR,D=1,CENA,CCFZ>       ; 2411
 1716  003122' 231400  020140
 1717  003123' 241240  000000                   MWORD   <JMAP,J=4000>                           ; 2412 error
 1718  003124' 000000  000040
 1719  003125' 241300  000000                   MWORD   <CONT,A=10,B=15,SOA,XOR,D=1>            ; 2413
 1720  003126' 461106  400340
 1721  003127' 241424  160000                   MWORD   <CJV,J=2416,SOQ,OR,D=1,CENA,CCFZ>       ; 2414
 1722  003130' 231400  020140
 1723  003131' 241540  000000                   MWORD   <JMAP,J=4000>                           ; 2415 error
 1724  003132' 000000  000040
 1725  003133' 241600  000000                   MWORD   <CONT,A=11,B=16,SOA,XOR,D=2>            ; 2416
 1726  003134' 462117  000340
 1727  003135' 241724  210000                   MWORD   <CJV,J=2421,SOQ,OR,D=1,CENA,CCFZ>       ; 2417
 1728  003136' 231400  020140
 1729  003137' 242040  000000                   MWORD   <JMAP,J=4000>                           ; 2420 error
 1730  003140' 000000  000040
 1731  003141' 242100  000000                   MWORD   <CONT,A=12,B=17,SOA,XOR,D=3>            ; 2421
 1732  003142' 463127  400340
 1733  003143' 242224  240000                   MWORD   <CJV,J=2424,SOQ,OR,D=1,CENA,CCFZ>       ; 2422
 1734  003144' 231400  020140
 1735  003145' 242340  000000                   MWORD   <JMAP,J=4000>                           ; 2423 error
 1736  003146' 000000  000040
 1737  003147' 242400  000000                   MWORD   <CONT,A=13,B=0,SOA,XOR,D=5>             ; 2424
 1738  003150' 465130  000340
 1739  003151' 242524  270000                   MWORD   <CJV,J=2427,SOQ,OR,D=1,CENA,CCFZ>       ; 2425
 1740  003152' 231400  020140
 1741  003153' 242640  000000                   MWORD   <JMAP,J=4000>                           ; 2426 error
 1742  003154' 000000  000040
 1743  003155' 242700  000000                   MWORD   <CONT,A=14,B=1,SOA,XOR,D=7>             ; 2427
 1744  003156' 467140  400340
 1745  003157' 243024  320000                   MWORD   <CJV,J=2432,SOQ,OR,D=1,CENA,CCFZ>       ; 2430
 1746  003160' 231400  020140
 1747  003161' 243140  000000                   MWORD   <JMAP,J=4000>                           ; 2431 error
 1748  003162' 000000  000040
 1749  003163' 243200  000000                   MWORD   <CONT,A=2,B=7,SDA,XOR,D=1>              ; 2432
 1750  003164' 561023  400340
 1751  003165' 243324  350000                   MWORD   <CJV,J=2435,SOQ,OR,D=1,CENA,CCFZ>       ; 2433
 1752  003166' 231400  020140
 1753  003167' 243440  000000                   MWORD   <JMAP,J=4000>                           ; 2434 error
 1754  003170' 000000  000040
 1755  003171' 243500  000000                   MWORD   <CONT,A=3,B=10,SDA,XOR,D=2>             ; 2435
 1756  003172' 562034  000340
 1757  003173' 243624  400000                   MWORD   <CJV,J=2440,SOQ,OR,D=1,CENA,CCFZ>       ; 2436
 1758  003174' 231400  020140
 1759  003175' 243740  000000                  MWORD   <JMAP,J=4000>                          ; 2437 error
 1760  003176' 000000  000040
 1761  003177' 244000  000000                  MWORD   <CONT,A=4,B=11,SDA,XOR,D=3>            ; 2440
 1762  003200' 563044  400340
 1763  003201' 244124  430000                  MWORD   <CJV,J=2443,SOQ,OR,D=1,CENA,CCFZ>      ; 2441
 1764  003202' 231400  020140
 1765  003203' 244240  000000                  MWORD   <JMAP,J=4000>                          ; 2442 error
 1766  003204' 000000  000040
 1767  003205' 244300  000000                  MWORD   <CONT,A=5,B=12,SDA,XOR,D=5>            ; 2443
 1768  003206' 565055  000340
 1769  003207' 244424  460000                  MWORD   <CJV,J=2446,SOQ,OR,D=1,CENA,CCFZ>      ; 2444
 1770  003210' 231400  020140
 1771  003211' 244540  000000                  MWORD   <JMAP,J=4000>                          ; 2445 error
 1772  003212' 000000  000040
 1773  003213' 244600  000000                  MWORD   <CONT,A=6,B=13,SDA,XOR,D=7>            ; 2446
 1774  003214' 567065  400340
 1775  003215' 244724  510000                  MWORD   <CJV,J=2451,SOQ,OR,D=1,CENA,CCFZ>      ; 2447
 1776  003216' 231400  020140
 1777  003217' 245040  000000                  MWORD   <JMAP,J=4000>                          ; 2450 error
 1778  003220' 000000  000040
 1779  003221' 245100  000000                  MWORD   <CONT,A=14,B=1,SDQ,XOR,D=1>            ; 2451
 1780  003222' 661140  400340
 1781  003223' 245224  540000                  MWORD   <CJV,J=2454,SOQ,OR,D=1,CENA,CCFZ>      ; 2452
 1782  003224' 231400  020140
 1783  003225' 245340  000000                  MWORD   <JMAP,J=4000>                          ; 2453 error
 1784  003226' 000000  000040
 1785  003227' 245400  000000                  MWORD   <CONT,A=15,B=2,SDQ,XOR,D=2>            ; 2454
 1786  003230' 662151  000340
 1787  003231' 245524  570000                  MWORD   <CJV,J=2457,SOQ,OR,D=1,CENA,CCFZ>      ; 2455
 1788  003232' 231400  020140
 1789  003233' 245640  000000                  MWORD   <JMAP,J=4000>                          ; 2456 error
 1790  003234' 000000  000040
 1791  003235' 245700  000000                  MWORD   <CONT,A=16,B=3,SDQ,XOR,D=3>            ; 2457
 1792  003236' 663161  400340
 1793  003237' 246024  620000                  MWORD   <CJV,J=2462,SOQ,OR,D=1,CENA,CCFZ>      ; 2460
 1794  003240' 231400  020140
 1795  003241' 246140  000000                  MWORD   <JMAP,J=4000>                          ; 2461 error
 1796  003242' 000000  000040
 1797  003243' 246200  000000                  MWORD   <CONT,A=17,B=4,SDQ,XOR,D=5>            ; 2462
 1798  003244' 665172  000340
 1799  003245' 246324  650000                  MWORD   <CJV,J=2465,SOQ,OR,D=1,CENA,CCFZ>      ; 2463
 1800  003246' 231400  020140
 1801  003247' 246440  000000                  MWORD   <JMAP,J=4000>                          ; 2464 error
 1802  003250' 000000  000040
 1803  003251' 246500  000000                  MWORD   <CONT,A=0,B=5,SDQ,XOR,D=7>             ; 2465
 1804  003252' 667002  400340
 1805  003253' 246624  700000                  MWORD   <CJV,J=2470,SOQ,OR,D=1,CENA,CCFZ>      ; 2466
 1806  003254' 231400  020140
 1807  003255' 246740  000000                  MWORD   <JMAP,J=4000>                          ; 2467 error
 1808  003256' 000000  000040
 1809  003257' 247000  000000                  MWORD   <CONT,A=6,B=13,SDQ,XOR,D=1>            ; 2470
 1810  003260' 761065  400340
 1811  003261' 247124  730000                  MWORD   <CJV,J=2473,SOQ,OR,D=1,CENA,CCFZ>      ; 2471
 1812  003262' 231400  020140
 1813  003263' 247240  000000                  MWORD   <JMAP,J=4000>                          ; 2472 error
```

```
003264' 000000  000040
003265' 247300  000000                 MWORD   <CONT,A=7,B=14,SDO,XOR,D=2>              ; 2473
003266' 762076  000340
003267' 247424  760000                 MWORD   <CJV,J=2476,SOQ,OR,D=1,CENA,CCFZ>        ; 2474
003270' 231400  020140
003271' 247540  000000                 MWORD   <JMAP,J=4000>                            ; 2475 error
003272' 000000  000040
003273' 247600  000000                 MWORD   <CONT,A=10,B=15,SDO,XOR,D=3>             ; 2476
003274' 763106  400340
003275' 247725  010000                 MWORD   <CJV,J=2501,SOQ,OR,D=1,CENA,CCFZ>        ; 2477
003276' 231400  020140
003277' 250040  000000                 MWORD   <JMAP,J=4000>                            ; 2500 error
003300' 000000  000040
003301' 250100  000000                 MWORD   <CONT,A=11,B=16,SDO,XOR,D=5>             ; 2501
003302' 765117  000340
003303' 250225  040000                 MWORD   <CJV,J=2504,SOQ,OR,D=1,CENA,CCFZ>        ; 2502
003304' 231400  020140
003305' 250340  000000                 MWORD   <JMAP,J=4000>                            ; 2503 error
003306' 000000  000040
003307' 250400  000000                 MWORD   <CONT,A=12,B=17,SDO,XOR,D=7>             ; 2504
003310' 767127  400340
003311' 250525  070000                 MWORD   <CJV,J=2507,SOQ,OR,D=1,CENA,CCFZ>        ; 2505
003312' 231400  020140
003313' 250640  000000                 MWORD   <JMAP,J=4000>                            ; 2506 error
003314' 000000  000040
003315' 250700  000000                 MWORD   <CONT,A=0,B=5,SAQ,XNOR,D=1>              ; 2507
003316' 071002  400340
003317' 251025  120000                 MWORD   <CJV,J=2512,SOQ,OR,D=1,CENA,CCFZ>        ; 2510
003320' 231400  020140
003321' 251140  000000                 MWORD   <JMAP,J=4000>                            ; 2511 error
003322' 000000  000040
003323' 251200  000000                 MWORD   <CONT,A=1,B=6,SAQ,XNOR,D=2>              ; 2512
003324' 072013  000340
003325' 251325  150000                 MWORD   <CJV,J=2515,SOQ,OR,D=1,CENA,CCFZ>        ; 2513
003326' 231400  020140
003327' 251440  000000                 MWORD   <JMAP,J=4000>                            ; 2514 error
003330' 000000  000040
003331' 251500  000000                 MWORD   <CONT,A=2,B=7,SAQ,XNOR,D=3>              ; 2515
003332' 073023  400340
003333' 251625  200000                 MWORD   <CJV,J=2520,SOQ,OR,D=1,CENA,CCFZ>        ; 2516
003334' 231400  020140
003335' 251740  000000                 MWORD   <JMAP,J=4000>                            ; 2517 error
003336' 000000  000040
003337' 252000  000000                 MWORD   <CONT,A=3,B=10,SAQ,XNOR,D=5>             ; 2520
003340' 075034  000340
003341' 252125  230000                 MWORD   <CJV,J=2523,SOQ,OR,D=1,CENA,CCFZ>        ; 2521
003342' 231400  020140
003343' 252240  000000                 MWORD   <JMAP,J=4000>                            ; 2522 error
003344' 000000  000040
003345' 252300  000000                 MWORD   <CONT,A=4,B=11,SAQ,XNOR,D=7>             ; 2523
003346' 077044  400340
003347' 252425  260000                 MWORD   <CJV,J=2526,SOQ,OR,D=1,CENA,CCFZ>        ; 2524
003350' 231400  020140
003351' 252540  000000                 MWORD   <JMAP,J=4000>                            ; 2525 error
003352' 000000  000040
```

```
  1869  003353' 252600  000000              MWORD   <CONT,A=12,B=17,SAB,XNOR,D=1>        ; 2526
  1870  003354' 171127  400340
  1871  003355' 252725  310000              MWORD   <CJV,J=2531,SOQ,OR,D=1,CENA,CCFZ>    ; 2527
  1872  003356' 231400  020140
  1873  003357' 253040  000000              MWORD   <JMAP,J=4000>                        ; 2530 error
  1874  003360' 000000  000040
  1875  003361' 253100  000000              MWORD   <CONT,A=13,B=0,SAB,XNOR,D=2>         ; 2531
  1876  003362' 172130  000340
  1877  003363' 253225  340000              MWORD   <CJV,J=2534,SOQ,OR,D=1,CENA,CCFZ>    ; 2532
  1878  003364' 231400  020140
  1879  003365' 253340  000000              MWORD   <JMAP,J=4000>                        ; 2533 error
  1880  003366' 000000  000040
  1881  003367' 253400  000000              MWORD   <CONT,A=14,B=1,SAB,XNOR,D=3>         ; 2534
  1882  003370' 173140  400340
  1883  003371' 253525  370000              MWORD   <CJV,J=2537,SOQ,OR,D=1,CENA,CCFZ>    ; 2535
  1884  003372' 231400  020140
  1885  003373' 253640  000000              MWORD   <JMAP,J=4000>                        ; 2536 error
  1886  003374' 000000  000040
  1887  003375' 253700  000000              MWORD   <CONT,A=15,B=2,SAB,XNOR,D=5>         ; 2537
  1888  003376' 175151  000340
  1889  003377' 254025  420000              MWORD   <CJV,J=2542,SOQ,OR,D=1,CENA,CCFZ>    ; 2540
  1890  003400' 231400  020140
  1891  003401' 254140  000000              MWORD   <JMAP,J=4000>                        ; 2541 error
  1892  003402' 000000  000040
  1893  003403' 254200  000000              MWORD   <CONT,A=16,B=3,SAB,XNOR,D=7>         ; 2542
  1894  003404' 177161  400340
  1895  003405' 254325  450000              MWORD   <CJV,J=2545,SOQ,OR,D=1,CENA,CCFZ>    ; 2543
  1896  003406' 231400  020140
  1897  003407' 254440  000000              MWORD   <JMAP,J=4000>                        ; 2544 error
  1898  003410' 000000  000040
  1899  003411' 254500  000000              MWORD   <CONT,A=4,B=11,SOQ,XNOR,D=1>         ; 2545
  1900  003412' 271044  400340
  1901  003413' 254625  500000              MWORD   <CJV,J=2550,SOQ,OR,D=1,CENA,CCFZ>    ; 2546
  1902  003414' 231400  020140
  1903  003415' 254740  000000              MWORD   <JMAP,J=4000>                        ; 2547 error
  1904  003416' 000000  000040
  1905  003417' 255000  000000              MWORD   <CONT,A=5,B=12,SOQ,XNOR,D=2>         ; 2550
  1906  003420' 272055  000340
  1907  003421' 255125  530000              MWORD   <CJV,J=2553,SOQ,OR,D=1,CENA,CCFZ>    ; 2551
  1908  003422' 231400  020140
  1909  003423' 255240  000000              MWORD   <JMAP,J=4000>                        ; 2552 error
  1910  003424' 000000  000040
  1911  003425' 255300  000000              MWORD   <CONT,A=6,B=13,SOQ,XNOR,D=3>         ; 2553
  1912  003426' 273065  400340
  1913  003427' 255425  560000              MWORD   <CJV,J=2556,SOQ,OR,D=1,CENA,CCFZ>    ; 2554
  1914  003430' 231400  020140
  1915  003431' 255540  000000              MWORD   <JMAP,J=4000>                        ; 2555 error
  1916  003432' 000000  000040
  1917  003433' 255600  000000              MWORD   <CONT,A=7,B=14,SOQ,XNOR,D=5>         ; 2556
  1918  003434' 275076  000340
  1919  003435' 255725  610000              MWORD   <CJV,J=2561,SOQ,OR,D=1,CENA,CCFZ>    ; 2557
  1920  003436' 231400  020140
  1921  003437' 256040  000000              MWORD   <JMAP,J=4000>                        ; 2560 error
  1922  003440' 000000  000040
  1923  003441' 256100  000000              MWORD   <CONT,A=10,B=15,SOQ,XNOR,D=7>        ; 2561
```

```
003442' 277106  400340
003443' 256225  640000                  MWORD   <CJV,J=2564,SOQ,OR,D=1,CENA,CCFZ>      ; 2562
003444' 231400  020140
003445' 256340  000000                  MWORD   <JMAP,J=4000>                          ; 2563 error
003446' 000000  000040
003447' 256400  000000                  MWORD   <CONT,A=16,B=3,SOB,XNOR,D=1>           ; 2564
003450' 371161  400340
003451' 256525  670000                  MWORD   <CJV,J=2567,SOQ,OR,D=1,CENA,CCFZ>      ; 2565
003452' 231400  020140
003453' 256640  000000                  MWORD   <JMAP,J=4000>                          ; 2566 error
003454' 000000  000040
003455' 256700  000000                  MWORD   <CONT,A=17,B=4,SOB,XNOR,D=2>           ; 2567
003456' 372172  000340
003457' 257025  720000                  MWORD   <CJV,J=2572,SOQ,OR,D=1,CENA,CCFZ>      ; 2570
003460' 231400  020140
003461' 257140  000000                  MWORD   <JMAP,J=4000>                          ; 2571 error
003462' 000000  000040
003463' 257200  000000                  MWORD   <CONT,A=0,B=5,SOB,XNOR,D=3>            ; 2572
003464' 373002  400340
003465' 257325  750000                  MWORD   <CJV,J=2575,SOQ,OR,D=1,CENA,CCFZ>      ; 2573
003466' 231400  020140
003467' 257440  000000                  MWORD   <JMAP,J=4000>                          ; 2574 error
003470' 000000  000040
003471' 257500  000000                  MWORD   <CONT,A=1,B=6,SOB,XNOR,D=5>            ; 2575
003472' 375013  000340
003473' 257626  000000                  MWORD   <CJV,J=2600,SOQ,OR,D=1,CENA,CCFZ>      ; 2576
003474' 231400  020140
003475' 257740  000000                  MWORD   <JMAP,J=4000>                          ; 2577 error
003476' 000000  000040
003477' 260000  000000                  MWORD   <CONT,A=2,B=7,SOB,XNOR,D=7>            ; 2600
003500' 377023  400340
003501' 260126  030000                  MWORD   <CJV,J=2603,SOQ,OR,D=1,CENA,CCFZ>      ; 2601
003502' 231400  020140
003503' 260240  000000                  MWORD   <JMAP,J=4000>                          ; 2602 error
003504' 000000  000040
003505' 260300  000000                  MWORD   <CONT,A=10,B=15,SOA,XNOR,D=1>          ; 2603
003506' 471106  400340
003507' 260426  060000                  MWORD   <CJV,J=2606,SOQ,OR,D=1,CENA,CCFZ>      ; 2604
003510' 231400  020140
003511' 260540  000000                  MWORD   <JMAP,J=4000>                          ; 2605 error
003512' 000000  000040
003513' 260600  000000                  MWORD   <CONT,A=11,B=16,SOA,XNOR,D=2>          ; 2606
003514' 472117  000340
003515' 260726  110000                  MWORD   <CJV,J=2611,SOQ,OR,D=1,CENA,CCFZ>      ; 2607
003516' 231400  020140
003517' 261040  000000                  MWORD   <JMAP,J=4000>                          ; 2610 error
003520' 000000  000040
003521' 261100  000000                  MWORD   <CONT,A=12,B=17,SOA,XNOR,D=3>          ; 2611
003522' 473127  400340
003523' 261226  140000                  MWORD   <CJV,J=2614,SOQ,OR,D=1,CENA,CCFZ>      ; 2612
003524' 231400  020140
003525' 261340  000000                  MWORD   <JMAP,J=4000>                          ; 2613 error
003526' 000000  000040
003527' 261400  000000                  MWORD   <CONT,A=13,B=0,SOA,XNOR.D=5>           ; 2614
003530' 475130  000340
```

```
003531' 261526  170000          MWORD   <CJV,J=2617,SOQ,OR,D=1,CENA,CCFZ>       ; 2615
003532' 231400  020140
003533' 261640  000000          MWORD   <JMAP,J=4000>                           ; 2616 error
003534' 000000  000040
003535' 261700  000000          MWORD   <CONT,A=14,B=1,SOA,XNOR,D=7>            ; 2617
003536' 477140  400340
003537' 262026  220000          MWORD   <CJV,J=2622,SOQ,OR,D=1,CENA,CCFZ>       ; 2620
003540' 231400  020140
003541' 262140  000000          MWORD   <JMAP,J=4000>                           ; 2621 error
003542' 000000  000040
003543' 262200  000000          MWORD   <CONT,A=2,B=7,SDA,XNOR,D=1>             ; 2622
003544' 571023  400340
003545' 262326  250000          MWORD   <CJV,J=2625,SOQ,OR,D=1,CENA,CCFZ>       ; 2623
003546' 231400  020140
003547' 262440  000000          MWORD   <JMAP,J=4000>                           ; 2624 error
003550' 000000  000040
003551' 262500  000000          MWORD   <CONT,A=3,B=10,SDA,XNOR,D=2>            ; 2625
003552' 572034  000340
003553' 262626  300000          MWORD   <CJV,J=2630,SOQ,OR,D=1,CENA,CCFZ>       ; 2626
003554' 231400  020140
003555' 262740  000000          MWORD   <JMAP,J=4000>                           ; 2627 error
003556' 000000  000040
003557' 263000  000000          MWORD   <CONT,A=4,B=11,SDA,XNOR,D=3>            ; 2630
003560' 573044  400340
003561' 263126  330000          MWORD   <CJV,J=2633,SOQ,OR,D=1,CENA,CCFZ>       ; 2631
003562' 231400  020140
003563' 263240  000000          MWORD   <JMAP,J=4000>                           ; 2632 error
003564' 000000  000040
003565' 263300  000000          MWORD   <CONT,A=5,B=12,SDA,XNOR,D=5>            ; 2633
003566' 575055  000340
003567' 263426  360000          MWORD   <CJV,J=2636,SOQ,OR,D=1,CENA,CCFZ>       ; 2634
003570' 231400  020140
003571' 263540  000000          MWORD   <JMAP,J=4000>                           ; 2635 error
003572' 000000  000040
003573' 263600  000000          MWORD   <CONT,A=6,B=13,SDA,XNOR,D=7>            ; 2636
003574' 577065  400340
003575' 263726  410000          MWORD   <CJV,J=2641,SOQ,OR,D=1,CENA,CCFZ>       ; 2637
003576' 231400  020140
003577' 264040  000000          MWORD   <JMAP,J=4000>                           ; 2640 error
003600' 000000  000040
003601' 264100  000000          MWORD   <CONT,A=14,B=1,SDQ,XNOR,D=1>            ; 2641
003602' 671140  400340
003603' 264226  440000          MWORD   <CJV,J=2644,SOQ,OR,D=1,CENA,CCFZ>       ; 2642
003604' 231400  020140
003605' 264340  000000          MWORD   <JMAP,J=4000>                           ; 2643 error
003606' 000000  000040
003607' 264400  000000          MWORD   <CONT,A=15,B=2,SDQ,XNOR,D=2>            ; 2644
003610' 672151  000340
003611' 264526  470000          MWORD   <CJV,J=2647,SOQ,OR,D=1,CENA,CCFZ>       ; 2645
003612' 231400  020140
003613' 264640  000000          MWORD   <JMAP,J=4000>                           ; 2646 error
003614' 000000  000040
003615' 264700  000000          MWORD   <CONT,A=16,B=3,SDQ,XNOR,D=3>            ; 2647
003616' 673161  400340
003617' 265026  520000          MWORD   <CJV,J=2652,SOQ,OR,D=1,CENA,CCFZ>       ; 2650
```

```
2034  003620' 231400  020140
2035  003621' 265140  000000                MWORD   <JMAP,J=4000>                           ; 2651 error
2036  003622' 000000  000040
2037  003623' 265200  000000                MWORD   <CONT,A=17,B=4,SDQ,XNOR,D=5>            ; 2652
2038  003624' 675172  000340
2039  003625' 265326  550000                MWORD   <CJV,J=2655,SOQ,OR,D=1,CENA,CCFZ>       ; 2653
2040  003626' 231400  020140
2041  003627' 265440  000000                MWORD   <JMAP,J=4000>                           ; 2654 error
2042  003630' 000000  000040
2043  003631' 265500  000000                MWORD   <CONT,A=0,B=5,SDQ,XNOR,D=7>             ; 2655
2044  003632' 677002  400340
2045  003633' 265626  600000                MWORD   <CJV,J=2660,SOQ,OR,D=1,CENA,CCFZ>       ; 2656
2046  003634' 231400  020140
2047  003635' 265740  000000                MWORD   <JMAP,J=4000>                           ; 2657 error
2048  003636' 000000  000040
2049  003637' 266000  000000                MWORD   <CONT,A=6,B=13,SDO,XNOR,D=1>            ; 2660
2050  003640' 771065  400340
2051  003641' 266126  630000                MWORD   <CJV,J=2663,SOQ,OR,D=1,CENA,CCFZ>       ; 2661
2052  003642' 231400  020140
2053  003643' 266240  000000                MWORD   <JMAP,J=4000>                           ; 2662 error
2054  003644' 000000  000040
2055  003645' 266300  000000                MWORD   <CONT,A=7,B=14,SDO,XNOR,D=2>            ; 2663
2056  003646' 772076  000340
2057  003647' 266426  660000                MWORD   <CJV,J=2666,SOQ,OR,D=1,CENA,CCFZ>       ; 2664
2058  003650' 231400  020140
2059  003651' 266540  000000                MWORD   <JMAP,J=4000>                           ; 2665 error
2060  003652' 000000  000040
2061  003653' 266600  000000                MWORD   <CONT,A=10,B=15,SDO,XNOR,D=3>           ; 2666
2062  003654' 773106  400340
2063  003655' 266726  710000                MWORD   <CJV,J=2671,SOQ,OR,D=1,CENA,CCFZ>       ; 2667
2064  003656' 231400  020140
2065  003657' 267040  000000                MWORD   <JMAP,J=4000>                           ; 2670 error
2066  003660' 000000  000040
2067  003661' 267100  000000                MWORD   <CONT,A=11,B=16,SDO,XNOR,D=5>           ; 2671
2068  003662' 775117  000340
2069  003663' 267226  740000                MWORD   <CJV,J=2674,SOQ,OR,D=1,CENA,CCFZ>       ; 2672
2070  003664' 231400  020140
2071  003665' 267340  000000                MWORD   <JMAP,J=4000>                           ; 2673 error
2072  003666' 000000  000040
2073  003667' 267400  000000                MWORD   <CONT,A=12,B=17,SDO,XNOR,D=7>           ; 2674
2074  003670' 777127  400340
2075  003671' 267526  770000                MWORD   <CJV,J=2677,SOQ,OR,D=1,CENA,CCFZ>       ; 2675
2076  003672' 231400  020140
2077  003673' 267640  000000                MWORD   <JMAP,J=4000>                           ; 2676 error
2078  003674' 000000  000040
2079
2080                                ;       Done
2081
2082  003675' 267726  770000                MWORD   <JMAP,J=2677,BAD>                       ; 2677 done
2083  003676' 000000  000041
2084
2085                                ;       Error
2086
2087  003677' 400040  000000                MWORD   <ADDR=4000,JMAP,J=4000,BAD>             ; 4000 error
2088  003700' 000000  000041
```

```
 003701' 777777  777777                -1


                                  ;#********************************************************************
                                  ; End of 2901 Tests (Part 2)
                                  ;#********************************************************************

                                               XLIST

NO ERRORS DETECTED

PROGRAM BREAK IS 003712
CPU TIME USED 06:14.832

199P CORE USED
```

```
DFPTA5   MACRO %53B(1242) 08:26  1-Jul-85 Page S-1
DFPTA5   MAC      18-Apr-85 10:59          SYMBOL TABLE

AAPNT            000000   ext    .LSOA            000000   spd    .RRMIN   020000  000000   spd
AEXEC            000024'  ext    .LSOB            000000   spd    .RSOA    400000  000000   spd
ALU      020000  000000   spd    .LSOQ            000000   spd    .RSOB    300000  000000   spd
CALL     200000  000000   spd    .LSAB            000000   spd    .RSOQ    200000  000000   spd
E23              000027   spd    .LSAQ            000000   spd    .RSAB    100000  000000   spd
EBUS     400000  000000   spd    .LSDO            000000   spd    .RSAQ            000000   spd
ERFLG            000015          .LSDA            000000   spd    .RSDO    700000  000000   spd
GO       260740  000000          .LSDQ            000000   spd    .RSDA    500000  000000   spd
IPACLR           000000   ext    .LSKCN           000000   spd    .RSDQ    600000  000000   spd
LAST     010000  000000   spd    .LSMIN           000000   spd    .RSKCN           240000   spd
MA31             000032'         .LXNOR           000000   spd    .RSMIN   010000  000000   spd
MLAST    400000  000000   spd    .LXOR            000000   spd    .RXNOR   070000  000000   spd
NDMP     000400  000000   spd    .MA              000017   spd    .RXOR    060000  000000   spd
RTN      263740  000000          .MAND            000007   spd
SCOPER   027000  000000          .MB              000017   spd
T31M             000033'         .MBAD            000001   spd
TA31             000017'         .MCCFZ           000037   spd
TG31             000012'         .MCENA           000001   spd
TLOAD            000015'  ext    .MCJV            000017   spd
TRACE            000013'  ext    .MCONT           000017   spd
TSTA31           000000'  ent    .MD              000007   spd
TSTA32           000000   ext    .MJ              007777   spd
TSTA36           000000   ext    .MJMAP           000017   spd
TSTA37           000000   ext    .MMGC            001777   spd
TSTA40           000000   ext    .MNAND           000007   spd
TSTA41           000000   ext    .MOR             000007   spd
TSTSUB           000000   ext    .MPLUS           000007   spd
TX31             000031'         .MRMIN           000007   spd
TXALL    060000  000000   spd    .MSOA            000007   spd
Z5               000000'         .MSOB            000007   spd
ZALU     000002  000000   spd    .MSOQ            000007   spd
$ARG2            000001          .MSAB            000007   spd
$B               000044          .MSAQ            000007   spd
$CHR             424144          .MSDO            000007   spd
$GARG            000000          .MSDA            000007   spd
%ADDR            004001   spd    .MSDQ            000007   spd
%ML      400040  000000   spd    .MSKCN           000037   spd
%MR              000041   spd    .MSMIN           000007   spd
 .LA             000000   spd    .MXNOR           000007   spd
 .LADDR  000100  000000   spd    .MXOR            000007   spd
 .LAND           000000   spd    .RA      000010  000000   spd
 .LB             000000   spd    .RAND    040000  000000   spd
 .LBAD           000000   spd    .RB              400000   spd
 .LCCFZ          000000   spd    .RBAD            000001   spd
 .LCENA          000000   spd    .RCCFZ           020000   spd
 .LCJV           000000   spd    .RCENA   000400  000000   spd
 .LCONT          000000   spd    .RCJV            000140   spd
 .LD             000000   spd    .RCONT           000340   spd
 .LJ             010000   spd    .RD      001000  000000   spd
 .LJMAP          000000   spd    .RJ              000000   spd
 .LMGC           000001   spd    .RJMAP           000040   spd
 .LNAND          000000   spd    .RMGC            000000   spd
 .LOR            000000   spd    .RNAND   050000  000000   spd
 .LPLUS          000000   spd    .ROR     030000  000000   spd
 .LRMIN          000000   spd    .RPLUS           000000   spd
```

```
AAPNT      18#   108
AEXEC      18#    96
ALU        71
CALL      108
E23        73
EBUS       71
ERFLG      91     97
IPACLR     22#
LAST      108
MA31       98    108#
MLAST      73
NDMP       71
T31M       72     85     112#
TA31       91#    99
TG31       70     83#
TLOAD      18#    86
TRACE      18#    84
TSTA31     10     70#
TSTA32     14#    74
TSTA36     14#    75
TSTA37     14#    76
TSTA40     14#    77
TSTA41     14#    78
TSTSUB     18#
TX31      100    104#
TXALL     108
Z5         29#    83
ZALU       71
$ARG2     116#   116    121#   121    123#   123    125#   125    127#   127    129#   129    131#   131
          133#   133    135#   135    137#   137    139#   139    141#   141    143#   143    145#   145
          147#   147    149#   149    151#   151    153#   153    161#   161    163#   163    165#   165
          167#   167    169#   169    171#   171    173#   173    175#   175    177#   177    179#   179
          181#   181    183#   183    185#   185    187#   187    189#   189    191#   191    193#   193
          195#   195    197#   197    199#   199    201#   201    203#   203    205#   205    207#   207
          209#   209    211#   211    213#   213    215#   215    217#   217    219#   219    221#   221
          223#   223    225#   225    227#   227    229#   229    231#   231    233#   233    235#   235
          237#   237    239#   239    241#   241    243#   243    245#   245    247#   247    249#   249
          251#   251    253#   253    255#   255    257#   257    259#   259    261#   261    263#   263
          265#   265    267#   267    269#   269    271#   271    273#   273    275#   275    277#   277
          279#   279    281#   281    283#   283    285#   285    287#   287    289#   289    291#   291
          293#   293    295#   295    297#   297    299#   299    301#   301    303#   303    305#   305
          307#   307    309#   309    311#   311    313#   313    315#   315    317#   317    319#   319
          321#   321    323#   323    325#   325    327#   327    329#   329    331#   331    333#   333
          335#   335    337#   337    339#   339    341#   341    343#   343    345#   345    347#   347
          349#   349    351#   351    353#   353    355#   355    357#   357    359#   359    361#   361
          363#   363    365#   365    367#   367    369#   369    371#   371    373#   373    375#   375
          377#   377    379#   379    381#   381    383#   383    385#   385    387#   387    389#   389
          391#   391    393#   393    395#   395    397#   397    399#   399    401#   401    403#   403
          405#   405    407#   407    409#   409    411#   411    413#   413    415#   415    417#   417
          419#   419    421#   421    423#   423    425#   425    427#   427    429#   429    431#   431
          433#   433    435#   435    437#   437    439#   439    441#   441    443#   443    445#   445
          447#   447    449#   449    451#   451    453#   453    455#   455    457#   457    459#   459
          461#   461    463#   463    465#   465    467#   467    469#   469    471#   471    473#   473
```

```
 475#  475   477#  477   479#  479   481#  481   483#  483   485#  485   487#  487
 489#  489   491#  491   493#  493   495#  495   497#  497   499#  499   501#  501
 503#  503   505#  505   507#  507   509#  509   511#  511   513#  513   515#  515
 517#  517   519#  519   521#  521   523#  523   525#  525   527#  527   529#  529
 531#  531   533#  533   535#  535   537#  537   539#  539   541#  541   543#  543
 545#  545   547#  547   549#  549   551#  551   553#  553   555#  555   557#  557
 559#  559   561#  561   563#  563   565#  565   567#  567   569#  569   571#  571
 573#  573   575#  575   577#  577   579#  579   581#  581   583#  583   585#  585
 587#  587   589#  589   591#  591   593#  593   595#  595   597#  597   599#  599
 601#  601   603#  603   605#  605   607#  607   609#  609   611#  611   613#  613
 615#  615   617#  617   619#  619   621#  621   623#  623   625#  625   627#  627
 629#  629   631#  631   633#  633   635#  635   637#  637   639#  639   641#  641
 643#  643   645#  645   647#  647   649#  649   651#  651   653#  653   655#  655
 657#  657   659#  659   661#  661   663#  663   665#  665   667#  667   669#  669
 671#  671   673#  673   675#  675   677#  677   679#  679   681#  681   683#  683
 685#  685   687#  687   689#  689   691#  691   693#  693   695#  695   697#  697
 699#  699   701#  701   703#  703   705#  705   707#  707   709#  709   711#  711
 713#  713   715#  715   717#  717   719#  719   721#  721   723#  723   725#  725
 727#  727   729#  729   731#  731   733#  733   735#  735   737#  737   739#  739
 741#  741   743#  743   745#  745   747#  747   749#  749   751#  751   753#  753
 755#  755   757#  757   759#  759   761#  761   763#  763   765#  765   767#  767
 769#  769   771#  771   773#  773   775#  775   777#  777   779#  779   781#  781
 783#  783   785#  785   787#  787   789#  789   791#  791   793#  793   795#  795
 797#  797   799#  799   801#  801   803#  803   805#  805   807#  807   809#  809
 811#  811   813#  813   815#  815   817#  817   819#  819   821#  821   823#  823
 825#  825   827#  827   829#  829   831#  831   833#  833   835#  835   837#  837
 839#  839   841#  841   843#  843   845#  845   847#  847   849#  849   851#  851
 853#  853   855#  855   857#  857   859#  859   861#  861   863#  863   865#  865
 867#  867   869#  869   871#  871   873#  873   875#  875   877#  877   879#  879
 881#  881   883#  883   885#  885   887#  887   889#  889   891#  891   893#  893
 895#  895   897#  897   899#  899   901#  901   903#  903   905#  905   907#  907
 909#  909   911#  911   913#  913   915#  915   917#  917   919#  919   921#  921
 923#  923   925#  925   927#  927   929#  929   931#  931   933#  933   935#  935
 937#  937   939#  939   941#  941   943#  943   945#  945   947#  947   949#  949
 951#  951   953#  953   955#  955   957#  957   959#  959   961#  961   963#  963
 965#  965   967#  967   969#  969   971#  971   973#  973   975#  975   977#  977
 979#  979   981#  981   983#  983   985#  985   987#  987   989#  989   991#  991
 993#  993   995#  995   997#  997   999#  999  1001# 1001  1003# 1003  1005# 1005
1007# 1007  1009# 1009  1011# 1011  1013# 1013  1015# 1015  1017# 1017  1019# 1019
1021# 1021  1023# 1023  1025# 1025  1027# 1027  1029# 1029  1031# 1031  1033# 1033
1035# 1035  1037# 1037  1039# 1039  1041# 1041  1043# 1043  1045# 1045  1047# 1047
1049# 1049  1051# 1051  1053# 1053  1055# 1055  1057# 1057  1059# 1059  1061# 1061
1063# 1063  1065# 1065  1067# 1067  1069# 1069  1071# 1071  1073# 1073  1075# 1075
1077# 1077  1079# 1079  1081# 1081  1083# 1083  1085# 1085  1087# 1087  1089# 1089
1091# 1091  1093# 1093  1095# 1095  1097# 1097  1099# 1099  1101# 1101  1103# 1103
1105# 1105  1107# 1107  1109# 1109  1111# 1111  1113# 1113  1115# 1115  1117# 1117
1119# 1119  1121# 1121  1123# 1123  1125# 1125  1127# 1127  1129# 1129  1131# 1131
1133# 1133  1135# 1135  1137# 1137  1139# 1139  1141# 1141  1143# 1143  1145# 1145
1147# 1147  1149# 1149  1151# 1151  1153# 1153  1155# 1155  1157# 1157  1159# 1159
1161# 1161  1163# 1163  1165# 1165  1167# 1167  1169# 1169  1171# 1171  1173# 1173
1175# 1175  1177# 1177  1179# 1179  1181# 1181  1183# 1183  1185# 1185  1187# 1187
1189# 1189  1191# 1191  1193# 1193  1195# 1195  1197# 1197  1199# 1199  1201# 1201
1203# 1203  1205# 1205  1207# 1207  1209# 1209  1211# 1211  1213# 1213  1215# 1215
```

```
1217#  1217   1219#  1219   1221#  1221   1223#  1223   1225#  1225   1227#  1227   1229#  1229
1231#  1231   1233#  1233   1235#  1235   1237#  1237   1239#  1239   1241#  1241   1243#  1243
1245#  1245   1247#  1247   1249#  1249   1251#  1251   1253#  1253   1255#  1255   1257#  1257
1259#  1259   1261#  1261   1263#  1263   1265#  1265   1267#  1267   1269#  1269   1271#  1271
1273#  1273   1275#  1275   1277#  1277   1279#  1279   1281#  1281   1283#  1283   1285#  1285
1287#  1287   1289#  1289   1291#  1291   1293#  1293   1295#  1295   1297#  1297   1299#  1299
1301#  1301   1303#  1303   1305#  1305   1307#  1307   1309#  1309   1311#  1311   1313#  1313
1315#  1315   1317#  1317   1319#  1319   1321#  1321   1323#  1323   1325#  1325   1327#  1327
1329#  1329   1331#  1331   1333#  1333   1335#  1335   1337#  1337   1339#  1339   1341#  1341
1343#  1343   1345#  1345   1347#  1347   1349#  1349   1351#  1351   1353#  1353   1355#  1355
1357#  1357   1359#  1359   1361#  1361   1363#  1363   1365#  1365   1367#  1367   1369#  1369
1371#  1371   1373#  1373   1375#  1375   1377#  1377   1379#  1379   1381#  1381   1383#  1383
1385#  1385   1387#  1387   1389#  1389   1391#  1391   1393#  1393   1395#  1395   1397#  1397
1399#  1399   1401#  1401   1403#  1403   1405#  1405   1407#  1407   1409#  1409   1411#  1411
1413#  1413   1415#  1415   1417#  1417   1419#  1419   1421#  1421   1423#  1423   1425#  1425
1427#  1427   1429#  1429   1431#  1431   1433#  1433   1435#  1435   1437#  1437   1439#  1439
1441#  1441   1443#  1443   1445#  1445   1447#  1447   1449#  1449   1451#  1451   1453#  1453
1455#  1455   1457#  1457   1459#  1459   1461#  1461   1463#  1463   1465#  1465   1467#  1467
1469#  1469   1471#  1471   1473#  1473   1475#  1475   1477#  1477   1479#  1479   1481#  1481
1483#  1483   1485#  1485   1487#  1487   1489#  1489   1491#  1491   1493#  1493   1495#  1495
1497#  1497   1499#  1499   1501#  1501   1503#  1503   1505#  1505   1507#  1507   1509#  1509
1511#  1511   1513#  1513   1515#  1515   1517#  1517   1519#  1519   1521#  1521   1523#  1523
1525#  1525   1527#  1527   1529#  1529   1531#  1531   1533#  1533   1535#  1535   1537#  1537
1539#  1539   1541#  1541   1543#  1543   1545#  1545   1547#  1547   1549#  1549   1551#  1551
1553#  1553   1555#  1555   1557#  1557   1559#  1559   1561#  1561   1563#  1563   1565#  1565
1567#  1567   1569#  1569   1571#  1571   1573#  1573   1575#  1575   1577#  1577   1579#  1579
1581#  1581   1583#  1583   1585#  1585   1587#  1587   1589#  1589   1591#  1591   1593#  1593
1595#  1595   1597#  1597   1599#  1599   1601#  1601   1603#  1603   1605#  1605   1607#  1607
1609#  1609   1611#  1611   1613#  1613   1615#  1615   1617#  1617   1619#  1619   1621#  1621
1623#  1623   1625#  1625   1627#  1627   1629#  1629   1631#  1631   1633#  1633   1635#  1635
1637#  1637   1639#  1639   1641#  1641   1643#  1643   1645#  1645   1647#  1647   1649#  1649
1651#  1651   1653#  1653   1655#  1655   1657#  1657   1659#  1659   1661#  1661   1663#  1663
1665#  1665   1667#  1667   1669#  1669   1671#  1671   1673#  1673   1675#  1675   1677#  1677
1679#  1679   1681#  1681   1683#  1683   1685#  1685   1687#  1687   1689#  1689   1691#  1691
1693#  1693   1695#  1695   1697#  1697   1699#  1699   1701#  1701   1703#  1703   1705#  1705
1707#  1707   1709#  1709   1711#  1711   1713#  1713   1715#  1715   1717#  1717   1719#  1719
1721#  1721   1723#  1723   1725#  1725   1727#  1727   1729#  1729   1731#  1731   1733#  1733
1735#  1735   1737#  1737   1739#  1739   1741#  1741   1743#  1743   1745#  1745   1747#  1747
1749#  1749   1751#  1751   1753#  1753   1755#  1755   1757#  1757   1759#  1759   1761#  1761
1763#  1763   1765#  1765   1767#  1767   1769#  1769   1771#  1771   1773#  1773   1775#  1775
1777#  1777   1779#  1779   1781#  1781   1783#  1783   1785#  1785   1787#  1787   1789#  1789
1791#  1791   1793#  1793   1795#  1795   1797#  1797   1799#  1799   1801#  1801   1803#  1803
1805#  1805   1807#  1807   1809#  1809   1811#  1811   1813#  1813   1815#  1815   1817#  1817
1819#  1819   1821#  1821   1823#  1823   1825#  1825   1827#  1827   1829#  1829   1831#  1831
1833#  1833   1835#  1835   1837#  1837   1839#  1839   1841#  1841   1843#  1843   1845#  1845
1847#  1847   1849#  1849   1851#  1851   1853#  1853   1855#  1855   1857#  1857   1859#  1859
1861#  1861   1863#  1863   1865#  1865   1867#  1867   1869#  1869   1871#  1871   1873#  1673
1875#  1875   1877#  1877   1879#  1879   1881#  1881   1883#  1883   1885#  1885   1887#  1887
1889#  1889   1891#  1891   1893#  1893   1895#  1895   1897#  1897   1899#  1899   1901#  1901
1903#  1903   1905#  1905   1907#  1907   1909#  1909   1911#  1911   1913#  1913   1915#  1915
1917#  1917   1919#  1919   1921#  1921   1923#  1923   1925#  1925   1927#  1927   1929#  1929
1931#  1931   1933#  1933   1935#  1935   1937#  1937   1939#  1939   1941#  1941   1943#  1943
1945#  1945   1947#  1947   1949#  1949   1951#  1951   1953#  1953   1955#  1955   1957#  1957
```

```
1127  1959# 1959   1961# 1961   1963# 1963   1965# 1965   1967# 1967   1969# 1969   1971# 1971     SEQ 1140
      1973# 1973   1975# 1975   1977# 1977   1979# 1979   1981# 1981   1983# 1983   1985# 1985
      1987# 1987   1989# 1989   1991# 1991   1993# 1993   1995# 1995   1997# 1997   1999# 1999
      2001# 2001   2003# 2003   2005# 2005   2007# 2007   2009# 2009   2011# 2011   2013# 2013
      2015# 2015   2017# 2017   2019# 2019   2021# 2021   2023# 2023   2025# 2025   2027# 2027
      2029# 2029   2031# 2031   2033# 2033   2035# 2035   2037# 2037   2039# 2039   2041# 2041
      2043# 2043   2045# 2045   2047# 2047   2049# 2049   2051# 2051   2053# 2053   2055# 2055
      2057# 2057   2059# 2059   2061# 2061   2063# 2063   2065# 2065   2067# 2067   2069# 2069
      2071# 2071   2073# 2073   2075# 2075   2077# 2077   2082# 2082   2087# 2087
$B     116#  116    121#  121    123#  123    125#  125    127#  127    129#  129    131#  131
       133#  133    135#  135    137#  137    139#  139    141#  141    143#  143    145#  145
       147#  147    149#  149    151#  151    153#  153    161#  161    163#  163    165#  165
       167#  167    169#  169    171#  171    173#  173    175#  175    177#  177    179#  179
       181#  181    183#  183    185#  185    187#  187    189#  189    191#  191    193#  193
       195#  195    197#  197    199#  199    201#  201    203#  203    205#  205    207#  207
       209#  209    211#  211    213#  213    215#  215    217#  217    219#  219    221#  221
       223#  223    225#  225    227#  227    229#  229    231#  231    233#  233    235#  235
       237#  237    239#  239    241#  241    243#  243    245#  245    247#  247    249#  249
       251#  251    253#  253    255#  255    257#  257    259#  259    261#  261    263#  263
       265#  265    267#  267    269#  269    271#  271    273#  273    275#  275    277#  277
       279#  279    281#  281    283#  283    285#  285    287#  287    289#  289    291#  291
       293#  293    295#  295    297#  297    299#  299    301#  301    303#  303    305#  305
       307#  307    309#  309    311#  311    313#  313    315#  315    317#  317    319#  319
       321#  321    323#  323    325#  325    327#  327    329#  329    331#  331    333#  333
       335#  335    337#  337    339#  339    341#  341    343#  343    345#  345    347#  347
       349#  349    351#  351    353#  353    355#  355    357#  357    359#  359    361#  361
       363#  363    365#  365    367#  367    369#  369    371#  371    373#  373    375#  375
       377#  377    379#  379    381#  381    383#  383    385#  385    387#  387    389#  389
       391#  391    393#  393    395#  395    397#  397    399#  399    401#  401    403#  403
       405#  405    407#  407    409#  409    411#  411    413#  413    415#  415    417#  417
       419#  419    421#  421    423#  423    425#  425    427#  427    429#  429    431#  431
       433#  433    435#  435    437#  437    439#  439    441#  441    443#  443    445#  445
       447#  447    449#  449    451#  451    453#  453    455#  455    457#  457    459#  459
       461#  461    463#  463    465#  465    467#  467    469#  469    471#  471    473#  473
       475#  475    477#  477    479#  479    481#  481    483#  483    485#  485    487#  487
       489#  489    491#  491    493#  493    495#  495    497#  497    499#  499    501#  501
       503#  503    505#  505    507#  507    509#  509    511#  511    513#  513    515#  515
       517#  517    519#  519    521#  521    523#  523    525#  525    527#  527    529#  529
       531#  531    533#  533    535#  535    537#  537    539#  539    541#  541    543#  543
       545#  545    547#  547    549#  549    551#  551    553#  553    555#  555    557#  557
       559#  559    561#  561    563#  563    565#  565    567#  567    569#  569    571#  571
       573#  573    575#  575    577#  577    579#  579    581#  581    583#  583    585#  585
       587#  587    589#  589    591#  591    593#  593    595#  595    597#  597    599#  599
       601#  601    603#  603    605#  605    607#  607    609#  609    611#  611    613#  613
       615#  615    617#  617    619#  619    621#  621    623#  623    625#  625    627#  627
       629#  629    631#  631    633#  633    635#  635    637#  637    639#  639    641#  641
       643#  643    645#  645    647#  647    649#  649    651#  651    653#  653    655#  655
       657#  657    659#  659    661#  661    663#  663    665#  665    667#  667    669#  669
       671#  671    673#  673    675#  675    677#  677    679#  679    681#  681    683#  683
       685#  685    687#  687    689#  689    691#  691    693#  693    695#  695    697#  697
       699#  699    701#  701    703#  703    705#  705    707#  707    709#  709    711#  711
       713#  713    715#  715    717#  717    719#  719    721#  721    723#  723    725#  725
       727#  727    729#  729    731#  731    733#  733    735#  735    737#  737    739#  739
```

```
 741#  741   743#  743   745#  745   747#  747   749#  749   751#  751   753#  753
 755#  755   757#  757   759#  759   761#  761   763#  763   765#  765   767#  767
 769#  769   771#  771   773#  773   775#  775   777#  777   779#  779   781#  781
 783#  783   785#  785   787#  787   789#  789   791#  791   793#  793   795#  795
 797#  797   799#  799   801#  801   803#  803   805#  805   807#  807   809#  809
 811#  811   813#  813   815#  815   817#  817   819#  819   821#  821   823#  823
 825#  825   827#  827   829#  829   831#  831   833#  833   835#  835   837#  837
 839#  839   841#  841   843#  843   845#  845   847#  847   849#  849   851#  851
 853#  853   855#  855   857#  857   859#  859   861#  861   863#  863   865#  865
 867#  867   869#  869   871#  871   873#  873   875#  875   877#  877   879#  879
 881#  881   883#  883   885#  885   887#  887   889#  889   891#  891   893#  893
 895#  895   897#  897   899#  899   901#  901   903#  903   905#  905   907#  907
 909#  909   911#  911   913#  913   915#  915   917#  917   919#  919   921#  921
 923#  923   925#  925   927#  927   929#  929   931#  931   933#  933   935#  935
 937#  937   939#  939   941#  941   943#  943   945#  945   947#  947   949#  949
 951#  951   953#  953   955#  955   957#  957   959#  959   961#  961   963#  963
 965#  965   967#  967   969#  969   971#  971   973#  973   975#  975   977#  977
 979#  979   981#  981   983#  983   985#  985   987#  987   989#  989   991#  991
 993#  993   995#  995   997#  997   999#  999  1001# 1001  1003# 1003  1005# 1005
1007# 1007  1009# 1009  1011# 1011  1013# 1013  1015# 1015  1017# 1017  1019# 1019
1021# 1021  1023# 1023  1025# 1025  1027# 1027  1029# 1029  1031# 1031  1033# 1033
1035# 1035  1037# 1037  1039# 1039  1041# 1041  1043# 1043  1045# 1045  1047# 1047
1049# 1049  1051# 1051  1053# 1053  1055# 1055  1057# 1057  1059# 1059  1061# 1061
1063# 1063  1065# 1065  1067# 1067  1069# 1069  1071# 1071  1073# 1073  1075# 1075
1077# 1077  1079# 1079  1081# 1081  1083# 1083  1085# 1085  1087# 1087  1089# 1089
1091# 1091  1093# 1093  1095# 1095  1097# 1097  1099# 1099  1101# 1101  1103# 1103
1105# 1105  1107# 1107  1109# 1109  1111# 1111  1113# 1113  1115# 1115  1117# 1117
1119# 1119  1121# 1121  1123# 1123  1125# 1125  1127# 1127  1129# 1129  1131# 1131
1133# 1133  1135# 1135  1137# 1137  1139# 1139  1141# 1141  1143# 1143  1145# 1145
1147# 1147  1149# 1149  1151# 1151  1153# 1153  1155# 1155  1157# 1157  1159# 1159
1161# 1161  1163# 1163  1165# 1165  1167# 1167  1169# 1169  1171# 1171  1173# 1173
1175# 1175  1177# 1177  1179# 1179  1181# 1181  1183# 1183  1185# 1185  1187# 1187
1189# 1189  1191# 1191  1193# 1193  1195# 1195  1197# 1197  1199# 1199  1201# 1201
1203# 1203  1205# 1205  1207# 1207  1209# 1209  1211# 1211  1213# 1213  1215# 1215
1217# 1217  1219# 1219  1221# 1221  1223# 1223  1225# 1225  1227# 1227  1229# 1229
1231# 1231  1233# 1233  1235# 1235  1237# 1237  1239# 1239  1241# 1241  1243# 1243
1245# 1245  1247# 1247  1249# 1249  1251# 1251  1253# 1253  1255# 1255  1257# 1257
1259# 1259  1261# 1261  1263# 1263  1265# 1265  1267# 1267  1269# 1269  1271# 1271
1273# 1273  1275# 1275  1277# 1277  1279# 1279  1281# 1281  1283# 1283  1285# 1285
1287# 1287  1289# 1289  1291# 1291  1293# 1293  1295# 1295  1297# 1297  1299# 1299
1301# 1301  1303# 1303  1305# 1305  1307# 1307  1309# 1309  1311# 1311  1313# 1313
1315# 1315  1317# 1317  1319# 1319  1321# 1321  1323# 1323  1325# 1325  1327# 1327
1329# 1329  1331# 1331  1333# 1333  1335# 1335  1337# 1337  1339# 1339  1341# 1341
1343# 1343  1345# 1345  1347# 1347  1349# 1349  1351# 1351  1353# 1353  1355# 1355
1357# 1357  1359# 1359  1361# 1361  1363# 1363  1365# 1365  1367# 1367  1369# 1369
1371# 1371  1373# 1373  1375# 1375  1377# 1377  1379# 1379  1381# 1381  1383# 1383
1385# 1385  1387# 1387  1389# 1389  1391# 1391  1393# 1393  1395# 1395  1397# 1397
1399# 1399  1401# 1401  1403# 1403  1405# 1405  1407# 1407  1409# 1409  1411# 1411
1413# 1413  1415# 1415  1417# 1417  1419# 1419  1421# 1421  1423# 1423  1425# 1425
1427# 1427  1429# 1429  1431# 1431  1433# 1433  1435# 1435  1437# 1437  1439# 1439
1441# 1441  1443# 1443  1445# 1445  1447# 1447  1449# 1449  1451# 1451  1453# 1453
1455# 1455  1457# 1457  1459# 1459  1461# 1461  1463# 1463  1465# 1465  1467# 1467
1469# 1469  1471# 1471  1473# 1473  1475# 1475  1477# 1477  1479# 1479  1481# 1481
```

```
1129    1483#  1483    1485#  1485    1487#  1487    1489#  1489    1491#  1491    1493#  1493    1495#  1495
        1497#  1497    1499#  1499    1501#  1501    1503#  1503    1505#  1505    1507#  1507    1509#  1509     SEQ 1142
        1511#  1511    1513#  1513    1515#  1515    1517#  1517    1519#  1519    1521#  1521    1523#  1523
        1525#  1525    1527#  1527    1529#  1529    1531#  1531    1533#  1533    1535#  1535    1537#  1537
        1539#  1539    1541#  1541    1543#  1543    1545#  1545    1547#  1547    1549#  1549    1551#  1551
        1553#  1553    1555#  1555    1557#  1557    1559#  1559    1561#  1561    1563#  1563    1565#  1565
        1567#  1567    1569#  1569    1571#  1571    1573#  1573    1575#  1575    1577#  1577    1579#  1579
        1581#  1581    1583#  1583    1585#  1585    1587#  1587    1589#  1589    1591#  1591    1593#  1593
        1595#  1595    1597#  1597    1599#  1599    1601#  1601    1603#  1603    1605#  1605    1607#  1607
        1609#  1609    1611#  1611    1613#  1613    1615#  1615    1617#  1617    1619#  1619    1621#  1621
        1623#  1623    1625#  1625    1627#  1627    1629#  1629    1631#  1631    1633#  1633    1635#  1635
        1637#  1637    1639#  1639    1641#  1641    1643#  1643    1645#  1645    1647#  1647    1649#  1649
        1651#  1651    1653#  1653    1655#  1655    1657#  1657    1659#  1659    1661#  1661    1663#  1663
        1665#  1665    1667#  1667    1669#  1669    1671#  1671    1673#  1673    1675#  1675    1677#  1677
        1679#  1679    1681#  1681    1683#  1683    1685#  1685    1687#  1687    1689#  1689    1691#  1691
        1693#  1693    1695#  1695    1697#  1697    1699#  1699    1701#  1701    1703#  1703    1705#  1705
        1707#  1707    1709#  1709    1711#  1711    1713#  1713    1715#  1715    1717#  1717    1719#  1719
        1721#  1721    1723#  1723    1725#  1725    1727#  1727    1729#  1729    1731#  1731    1733#  1733
        1735#  1735    1737#  1737    1739#  1739    1741#  1741    1743#  1743    1745#  1745    1747#  1747
        1749#  1749    1751#  1751    1753#  1753    1755#  1755    1757#  1757    1759#  1759    1761#  1761
        1763#  1763    1765#  1765    1767#  1767    1769#  1769    1771#  1771    1773#  1773    1775#  1775
        1777#  1777    1779#  1779    1781#  1781    1783#  1783    1785#  1785    1787#  1787    1789#  1789
        1791#  1791    1793#  1793    1795#  1795    1797#  1797    1799#  1799    1801#  1801    1803#  1803
        1805#  1805    1807#  1807    1809#  1809    1811#  1811    1813#  1813    1815#  1815    1817#  1817
        1819#  1819    1821#  1821    1823#  1823    1825#  1825    1827#  1827    1829#  1829    1831#  1831
        1833#  1833    1835#  1835    1837#  1837    1839#  1839    1841#  1841    1843#  1843    1845#  1845
        1847#  1847    1849#  1849    1851#  1851    1853#  1853    1855#  1855    1857#  1857    1859#  1859
        1861#  1861    1863#  1863    1865#  1865    1867#  1867    1869#  1869    1871#  1871    1873#  1873
        1875#  1875    1877#  1877    1879#  1879    1881#  1881    1883#  1883    1885#  1885    1887#  1887
        1889#  1889    1891#  1891    1893#  1893    1895#  1895    1897#  1897    1899#  1899    1901#  1901
        1903#  1903    1905#  1905    1907#  1907    1909#  1909    1911#  1911    1913#  1913    1915#  1915
        1917#  1917    1919#  1919    1921#  1921    1923#  1923    1925#  1925    1927#  1927    1929#  1929
        1931#  1931    1933#  1933    1935#  1935    1937#  1937    1939#  1939    1941#  1941    1943#  1943
        1945#  1945    1947#  1947    1949#  1949    1951#  1951    1953#  1953    1955#  1955    1957#  1957
        1959#  1959    1961#  1961    1963#  1963    1965#  1965    1967#  1967    1969#  1969    1971#  1971
        1973#  1973    1975#  1975    1977#  1977    1979#  1979    1981#  1981    1983#  1983    1985#  1985
        1987#  1987    1989#  1989    1991#  1991    1993#  1993    1995#  1995    1997#  1997    1999#  1999
        2001#  2001    2003#  2003    2005#  2005    2007#  2007    2009#  2009    2011#  2011    2013#  2013
        2015#  2015    2017#  2017    2019#  2019    2021#  2021    2023#  2023    2025#  2025    2027#  2027
        2029#  2029    2031#  2031    2033#  2033    2035#  2035    2037#  2037    2039#  2039    2041#  2041
        2043#  2043    2045#  2045    2047#  2047    2049#  2049    2051#  2051    2053#  2053    2055#  2055
        2057#  2057    2059#  2059    2061#  2061    2063#  2063    2065#  2065    2067#  2067    2069#  2069
        2071#  2071    2073#  2073    2075#  2075    2077#  2077    2082#  2082    2087#  2087
$CHR     116#   116     121#   121     123#   123     125#   125     127#   127     129#   129     131#   131
         133#   133     135#   135     137#   137     139#   139     141#   141     143#   143     145#   145
         147#   147     149#   149     151#   151     153#   153     161#   161     163#   163     165#   165
         167#   167     169#   169     171#   171     173#   173     175#   175     177#   177     179#   179
         181#   181     183#   183     185#   185     187#   187     189#   189     191#   191     193#   193
         195#   195     197#   197     199#   199     201#   201     203#   203     205#   205     207#   207
         209#   209     211#   211     213#   213     215#   215     217#   217     219#   219     221#   221
         223#   223     225#   225     227#   227     229#   229     231#   231     233#   233     235#   235
         237#   237     239#   239     241#   241     243#   243     245#   245     247#   247     249#   249
         251#   251     253#   253     255#   255     257#   257     259#   259     261#   261     263#   263
```

```
Q 1130   265#  265    267#  267    269#  269    271#  271    273#  273    275#  275    277#  277    SEQ 1143
         279#  279    281#  281    283#  283    285#  285    287#  287    289#  289    291#  291
         293#  293    295#  295    297#  297    299#  299    301#  301    303#  303    305#  305
         307#  307    309#  309    311#  311    313#  313    315#  315    317#  317    319#  319
         321#  321    323#  323    325#  325    327#  327    329#  329    331#  331    333#  333
         335#  335    337#  337    339#  339    341#  341    343#  343    345#  345    347#  347
         349#  349    351#  351    353#  353    355#  355    357#  357    359#  359    361#  361
         363#  363    365#  365    367#  367    369#  369    371#  371    373#  373    375#  375
         377#  377    379#  379    381#  381    383#  383    385#  385    387#  387    389#  389
         391#  391    393#  393    395#  395    397#  397    399#  399    401#  401    403#  403
         405#  405    407#  407    409#  409    411#  411    413#  413    415#  415    417#  417
         419#  419    421#  421    423#  423    425#  425    427#  427    429#  429    431#  431
         433#  433    435#  435    437#  437    439#  439    441#  441    443#  443    445#  445
         447#  447    449#  449    451#  451    453#  453    455#  455    457#  457    459#  459
         461#  461    463#  463    465#  465    467#  467    469#  469    471#  471    473#  473
         475#  475    477#  477    479#  479    481#  481    483#  483    485#  485    487#  487
         489#  489    491#  491    493#  493    495#  495    497#  497    499#  499    501#  501
         503#  503    505#  505    507#  507    509#  509    511#  511    513#  513    515#  515
         517#  517    519#  519    521#  521    523#  523    525#  525    527#  527    529#  529
         531#  531    533#  533    535#  535    537#  537    539#  539    541#  541    543#  543
         545#  545    547#  547    549#  549    551#  551    553#  553    555#  555    557#  557
         559#  559    561#  561    563#  563    565#  565    567#  567    569#  569    571#  571
         573#  573    575#  575    577#  577    579#  579    581#  581    583#  583    585#  585
         587#  587    589#  589    591#  591    593#  593    595#  595    597#  597    599#  599
         601#  601    603#  603    605#  605    607#  607    609#  609    611#  611    613#  613
         615#  615    617#  617    619#  619    621#  621    623#  623    625#  625    627#  627
         629#  629    631#  631    633#  633    635#  635    637#  637    639#  639    641#  641
         643#  643    645#  645    647#  647    649#  649    651#  651    653#  653    655#  655
         657#  657    659#  659    661#  661    663#  663    665#  665    667#  667    669#  669
         671#  671    673#  673    675#  675    677#  677    679#  679    681#  681    683#  683
         685#  685    687#  687    689#  689    691#  691    693#  693    695#  695    697#  697
         699#  699    701#  701    703#  703    705#  705    707#  707    709#  709    711#  711
         713#  713    715#  715    717#  717    719#  719    721#  721    723#  723    725#  725
         727#  727    729#  729    731#  731    733#  733    735#  735    737#  737    739#  739
         741#  741    743#  743    745#  745    747#  747    749#  749    751#  751    753#  753
         755#  755    757#  757    759#  759    761#  761    763#  763    765#  765    767#  767
         769#  769    771#  771    773#  773    775#  775    777#  777    779#  779    781#  781
         783#  783    785#  785    787#  787    789#  789    791#  791    793#  793    795#  795
         797#  797    799#  799    801#  801    803#  803    805#  805    807#  807    809#  809
         811#  811    813#  813    815#  815    817#  817    819#  819    821#  821    823#  823
         825#  825    827#  827    829#  829    831#  831    833#  833    835#  835    837#  837
         839#  839    841#  841    843#  843    845#  845    847#  847    849#  849    851#  851
         853#  853    855#  855    857#  857    859#  859    861#  861    863#  863    865#  865
         867#  867    869#  869    871#  871    873#  873    875#  875    877#  877    879#  879
         881#  881    883#  883    885#  885    887#  887    889#  889    891#  891    893#  893
         895#  895    897#  897    899#  899    901#  901    903#  903    905#  905    907#  907
         909#  909    911#  911    913#  913    915#  915    917#  917    919#  919    921#  921
         923#  923    925#  925    927#  927    929#  929    931#  931    933#  933    935#  935
         937#  937    939#  939    941#  941    943#  943    945#  945    947#  947    949#  949
         951#  951    953#  953    955#  955    957#  957    959#  959    961#  961    963#  963
         965#  965    967#  967    969#  969    971#  971    973#  973    975#  975    977#  977
         979#  979    981#  981    983#  983    985#  985    987#  987    989#  989    991#  991
         993#  993    995#  995    997#  997    999#  999   1001# 1001   1003# 1003   1005# 1005
```

Q 1131

SEQ 1144

1007#	1007	1009#	1009	1011#	1011	1013#	1013	1015#	1015	1017#	1017	1019#	1019
1021#	1021	1023#	1023	1025#	1025	1027#	1027	1029#	1029	1031#	1031	1033#	1033
1035#	1035	1037#	1037	1039#	1039	1041#	1041	1043#	1043	1045#	1045	1047#	1047
1049#	1049	1051#	1051	1053#	1053	1055#	1055	1057#	1057	1059#	1059	1061#	1061
1063#	1063	1065#	1065	1067#	1067	1069#	1069	1071#	1071	1073#	1073	1075#	1075
1077#	1077	1079#	1079	1081#	1081	1083#	1083	1085#	1085	1087#	1087	1089#	1089
1091#	1091	1093#	1093	1095#	1095	1097#	1097	1099#	1099	1101#	1101	1103#	1103
1105#	1105	1107#	1107	1109#	1109	1111#	1111	1113#	1113	1115#	1115	1117#	1117
1119#	1119	1121#	1121	1123#	1123	1125#	1125	1127#	1127	1129#	1129	1131#	1131
1133#	1133	1135#	1135	1137#	1137	1139#	1139	1141#	1141	1143#	1143	1145#	1145
1147#	1147	1149#	1149	1151#	1151	1153#	1153	1155#	1155	1157#	1157	1159#	1159
1161#	1161	1163#	1163	1165#	1165	1167#	1167	1169#	1169	1171#	1171	1173#	1173
1175#	1175	1177#	1177	1179#	1179	1181#	1181	1183#	1183	1185#	1185	1187#	1187
1189#	1189	1191#	1191	1193#	1193	1195#	1195	1197#	1197	1199#	1199	1201#	1201
1203#	1203	1205#	1205	1207#	1207	1209#	1209	1211#	1211	1213#	1213	1215#	1215
1217#	1217	1219#	1219	1221#	1221	1223#	1223	1225#	1225	1227#	1227	1229#	1229
1231#	1231	1233#	1233	1235#	1235	1237#	1237	1239#	1239	1241#	1241	1243#	1243
1245#	1245	1247#	1247	1249#	1249	1251#	1251	1253#	1253	1255#	1255	1257#	1257
1259#	1259	1261#	1261	1263#	1263	1265#	1265	1267#	1267	1269#	1269	1271#	1271
1273#	1273	1275#	1275	1277#	1277	1279#	1279	1281#	1281	1283#	1283	1285#	1285
1287#	1287	1289#	1289	1291#	1291	1293#	1293	1295#	1295	1297#	1297	1299#	1299
1301#	1301	1303#	1303	1305#	1305	1307#	1307	1309#	1309	1311#	1311	1313#	1313
1315#	1315	1317#	1317	1319#	1319	1321#	1321	1323#	1323	1325#	1325	1327#	1327
1329#	1329	1331#	1331	1333#	1333	1335#	1335	1337#	1337	1339#	1339	1341#	1341
1343#	1343	1345#	1345	1347#	1347	1349#	1349	1351#	1351	1353#	1353	1355#	1355
1357#	1357	1359#	1359	1361#	1361	1363#	1363	1365#	1365	1367#	1367	1369#	1369
1371#	1371	1373#	1373	1375#	1375	1377#	1377	1379#	1379	1381#	1381	1383#	1383
1385#	1385	1387#	1387	1389#	1389	1391#	1391	1393#	1393	1395#	1395	1397#	1397
1399#	1399	1401#	1401	1403#	1403	1405#	1405	1407#	1407	1409#	1409	1411#	1411
1413#	1413	1415#	1415	1417#	1417	1419#	1419	1421#	1421	1423#	1423	1425#	1425
1427#	1427	1429#	1429	1431#	1431	1433#	1433	1435#	1435	1437#	1437	1439#	1439
1441#	1441	1443#	1443	1445#	1445	1447#	1447	1449#	1449	1451#	1451	1453#	1453
1455#	1455	1457#	1457	1459#	1459	1461#	1461	1463#	1463	1465#	1465	1467#	1467
1469#	1469	1471#	1471	1473#	1473	1475#	1475	1477#	1477	1479#	1479	1481#	1481
1483#	1483	1485#	1485	1487#	1487	1489#	1489	1491#	1491	1493#	1493	1495#	1495
1497#	1497	1499#	1499	1501#	1501	1503#	1503	1505#	1505	1507#	1507	1509#	1509
1511#	1511	1513#	1513	1515#	1515	1517#	1517	1519#	1519	1521#	1521	1523#	1523
1525#	1525	1527#	1527	1529#	1529	1531#	1531	1533#	1533	1535#	1535	1537#	1537
1539#	1539	1541#	1541	1543#	1543	1545#	1545	1547#	1547	1549#	1549	1551#	1551
1553#	1553	1555#	1555	1557#	1557	1559#	1559	1561#	1561	1563#	1563	1565#	1565
1567#	1567	1569#	1569	1571#	1571	1573#	1573	1575#	1575	1577#	1577	1579#	1579
1581#	1581	1583#	1583	1585#	1585	1587#	1587	1589#	1589	1591#	1591	1593#	1593
1595#	1595	1597#	1597	1599#	1599	1601#	1601	1603#	1603	1605#	1605	1607#	1607
1609#	1609	1611#	1611	1613#	1613	1615#	1615	1617#	1617	1619#	1619	1621#	1621
1623#	1623	1625#	1625	1627#	1627	1629#	1629	1631#	1631	1633#	1633	1635#	1635
1637#	1637	1639#	1639	1641#	1641	1643#	1643	1645#	1645	1647#	1647	1649#	1649
1651#	1651	1653#	1653	1655#	1655	1657#	1657	1659#	1659	1661#	1661	1663#	1663
1665#	1665	1667#	1667	1669#	1669	1671#	1671	1673#	1673	1675#	1675	1677#	1677
1679#	1679	1681#	1681	1683#	1683	1685#	1685	1687#	1687	1689#	1689	1691#	1691
1693#	1693	1695#	1695	1697#	1697	1699#	1699	1701#	1701	1703#	1703	1705#	1705
1707#	1707	1709#	1709	1711#	1711	1713#	1713	1715#	1715	1717#	1717	1719#	1719
1721#	1721	1723#	1723	1725#	1725	1727#	1727	1729#	1729	1731#	1731	1733#	1733
1735#	1735	1737#	1737	1739#	1739	1741#	1741	1743#	1743	1745#	1745	1747#	1747

```
                 1749#  1749    1751#  1751    1753#  1753    1755#  1755    1757#  1757    1759#  1759    1761#  1761      SEQ 1145
                 1763#  1763    1765#  1765    1767#  1767    1769#  1769    1771#  1771    1773#  1773    1775#  1775
                 1777#  1777    1779#  1779    1781#  1781    1783#  1783    1785#  1785    1787#  1787    1789#  1789
                 1791#  1791    1793#  1793    1795#  1795    1797#  1797    1799#  1799    1801#  1801    1803#  1803
                 1805#  1805    1807#  1807    1809#  1809    1811#  1811    1813#  1813    1815#  1815    1817#  1817
                 1819#  1819    1821#  1821    1823#  1823    1825#  1825    1827#  1827    1829#  1829    1831#  1831
                 1833#  1833    1835#  1835    1837#  1837    1839#  1839    1841#  1841    1843#  1843    1845#  1845
                 1847#  1847    1849#  1849    1851#  1851    1853#  1853    1855#  1855    1857#  1857    1859#  1859
                 1861#  1861    1863#  1863    1865#  1865    1867#  1867    1869#  1869    1871#  1871    1873#  1873
                 1875#  1875    1877#  1877    1879#  1879    1881#  1881    1883#  1883    1885#  1885    1887#  1887
                 1889#  1889    1891#  1891    1893#  1893    1895#  1895    1897#  1897    1899#  1899    1901#  1901
                 1903#  1903    1905#  1905    1907#  1907    1909#  1909    1911#  1911    1913#  1913    1915#  1915
                 1917#  1917    1919#  1919    1921#  1921    1923#  1923    1925#  1925    1927#  1927    1929#  1929
                 1931#  1931    1933#  1933    1935#  1935    1937#  1937    1939#  1939    1941#  1941    1943#  1943
                 1945#  1945    1947#  1947    1949#  1949    1951#  1951    1953#  1953    1955#  1955    1957#  1957
                 1959#  1959    1961#  1961    1963#  1963    1965#  1965    1967#  1967    1969#  1969    1971#  1971
                 1973#  1973    1975#  1975    1977#  1977    1979#  1979    1981#  1981    1983#  1983    1985#  1985
                 1987#  1987    1989#  1989    1991#  1991    1993#  1993    1995#  1995    1997#  1997    1999#  1999
                 2001#  2001    2003#  2003    2005#  2005    2007#  2007    2009#  2009    2011#  2011    2013#  2013
                 2015#  2015    2017#  2017    2019#  2019    2021#  2021    2023#  2023    2025#  2025    2027#  2027
                 2029#  2029    2031#  2031    2033#  2033    2035#  2035    2037#  2037    2039#  2039    2041#  2041
                 2043#  2043    2045#  2045    2047#  2047    2049#  2049    2051#  2051    2053#  2053    2055#  2055
                 2057#  2057    2059#  2059    2061#  2061    2063#  2063    2065#  2065    2067#  2067    2069#  2069
                 2071#  2071    2073#  2073    2075#  2075    2077#  2077    2082#  2082    2087#  2087
$GARG             116#   116     121#   121     123#   123     125#   125     127#   127     129#   129     131#   131
                  133#   133     135#   135     137#   137     139#   139     141#   141     143#   143     145#   145
                  147#   147     149#   149     151#   151     153#   153     161#   161     163#   163     165#   165
                  167#   167     169#   169     171#   171     173#   173     175#   175     177#   177     179#   179
                  181#   181     183#   183     185#   185     187#   187     189#   189     191#   191     193#   193
                  195#   195     197#   197     199#   199     201#   201     203#   203     205#   205     207#   207
                  209#   209     211#   211     213#   213     215#   215     217#   217     219#   219     221#   221
                  223#   223     225#   225     227#   227     229#   229     231#   231     233#   233     235#   235
                  237#   237     239#   239     241#   241     243#   243     245#   245     247#   247     249#   249
                  251#   251     253#   253     255#   255     257#   257     259#   259     261#   261     263#   263
                  265#   265     267#   267     269#   269     271#   271     273#   273     275#   275     277#   277
                  279#   279     281#   281     283#   283     285#   285     287#   287     289#   289     291#   291
                  293#   293     295#   295     297#   297     299#   299     301#   301     303#   303     305#   305
                  307#   307     309#   309     311#   311     313#   313     315#   315     317#   317     319#   319
                  321#   321     323#   323     325#   325     327#   327     329#   329     331#   331     333#   333
                  335#   335     337#   337     339#   339     341#   341     343#   343     345#   345     347#   347
                  349#   349     351#   351     353#   353     355#   355     357#   357     359#   359     361#   361
                  363#   363     365#   365     367#   367     369#   369     371#   371     373#   373     375#   375
                  377#   377     379#   379     381#   381     383#   383     385#   385     387#   387     389#   389
                  391#   391     393#   393     395#   395     397#   397     399#   399     401#   401     403#   403
                  405#   405     407#   407     409#   409     411#   411     413#   413     415#   415     417#   417
                  419#   419     421#   421     423#   423     425#   425     427#   427     429#   429     431#   431
                  433#   433     435#   435     437#   437     439#   439     441#   441     443#   443     445#   445
                  447#   447     449#   449     451#   451     453#   453     455#   455     457#   457     459#   459
                  461#   461     463#   463     465#   465     467#   467     469#   469     471#   471     473#   473
                  475#   475     477#   477     479#   479     481#   481     483#   483     485#   485     487#   487
                  489#   489     491#   491     493#   493     495#   495     497#   497     499#   499     501#   501
                  503#   503     505#   505     507#   507     509#   509     511#   511     513#   513     515#   515
                  517#   517     519#   519     521#   521     523#   523     525#   525     527#   527     529#   529
```

```
 531#  531    533#  533    535#  535    537#  537    539#  539    541#  541    543#  543      SEQ 1146
 545#  545    547#  547    549#  549    551#  551    553#  553    555#  555    557#  557
 559#  559    561#  561    563#  563    565#  565    567#  567    569#  569    571#  571
 573#  573    575#  575    577#  577    579#  579    581#  581    583#  583    585#  585
 587#  587    589#  589    591#  591    593#  593    595#  595    597#  597    599#  599
 601#  601    603#  603    605#  605    607#  607    609#  609    611#  611    613#  613
 615#  615    617#  617    619#  619    621#  621    623#  623    625#  625    627#  627
 629#  629    631#  631    633#  633    635#  635    637#  637    639#  639    641#  641
 643#  643    645#  645    647#  647    649#  649    651#  651    653#  653    655#  655
 657#  657    659#  659    661#  661    663#  663    665#  665    667#  667    669#  669
 671#  671    673#  673    675#  675    677#  677    679#  679    681#  681    683#  683
 685#  685    687#  687    689#  689    691#  691    693#  693    695#  695    697#  697
 699#  699    701#  701    703#  703    705#  705    707#  707    709#  709    711#  711
 713#  713    715#  715    717#  717    719#  719    721#  721    723#  723    725#  725
 727#  727    729#  729    731#  731    733#  733    735#  735    737#  737    739#  739
 741#  741    743#  743    745#  745    747#  747    749#  749    751#  751    753#  753
 755#  755    757#  757    759#  759    761#  761    763#  763    765#  765    767#  767
 769#  769    771#  771    773#  773    775#  775    777#  777    779#  779    781#  781
 783#  783    785#  785    787#  787    789#  789    791#  791    793#  793    795#  795
 797#  797    799#  799    801#  801    803#  803    805#  805    807#  807    809#  809
 811#  811    813#  813    815#  815    817#  817    819#  819    821#  821    823#  823
 825#  825    827#  827    829#  829    831#  831    833#  833    835#  835    837#  837
 839#  839    841#  841    843#  843    845#  845    847#  847    849#  849    851#  851
 853#  853    855#  855    857#  857    859#  859    861#  861    863#  863    865#  865
 867#  867    869#  869    871#  871    873#  873    875#  875    877#  877    879#  879
 881#  881    883#  883    885#  885    887#  887    889#  889    891#  891    893#  893
 895#  895    897#  897    899#  899    901#  901    903#  903    905#  905    907#  907
 909#  909    911#  911    913#  913    915#  915    917#  917    919#  919    921#  921
 923#  923    925#  925    927#  927    929#  929    931#  931    933#  933    935#  935
 937#  937    939#  939    941#  941    943#  943    945#  945    947#  947    949#  949
 951#  951    953#  953    955#  955    957#  957    959#  959    961#  961    963#  963
 965#  965    967#  967    969#  969    971#  971    973#  973    975#  975    977#  977
 979#  979    981#  981    983#  983    985#  985    987#  987    989#  989    991#  991
 993#  993    995#  995    997#  997    999#  999   1001# 1001   1003# 1003   1005# 1005
1007# 1007   1009# 1009   1011# 1011   1013# 1013   1015# 1015   1017# 1017   1019# 1019
1021# 1021   1023# 1023   1025# 1025   1027# 1027   1029# 1029   1031# 1031   1033# 1033
1035# 1035   1037# 1037   1039# 1039   1041# 1041   1043# 1043   1045# 1045   1047# 1047
1049# 1049   1051# 1051   1053# 1053   1055# 1055   1057# 1057   1059# 1059   1061# 1061
1063# 1063   1065# 1065   1067# 1067   1069# 1069   1071# 1071   1073# 1073   1075# 1075
1077# 1077   1079# 1079   1081# 1081   1083# 1083   1085# 1085   1087# 1087   1089# 1089
1091# 1091   1093# 1093   1095# 1095   1097# 1097   1099# 1099   1101# 1101   1103# 1103
1105# 1105   1107# 1107   1109# 1109   1111# 1111   1113# 1113   1115# 1115   1117# 1117
1119# 1119   1121# 1121   1123# 1123   1125# 1125   1127# 1127   1129# 1129   1131# 1131
1133# 1133   1135# 1135   1137# 1137   1139# 1139   1141# 1141   1143# 1143   1145# 1145
1147# 1147   1149# 1149   1151# 1151   1153# 1153   1155# 1155   1157# 1157   1159# 1159
1161# 1161   1163# 1163   1165# 1165   1167# 1167   1169# 1169   1171# 1171   1173# 1173
1175# 1175   1177# 1177   1179# 1179   1181# 1181   1183# 1183   1185# 1185   1187# 1187
1189# 1189   1191# 1191   1193# 1193   1195# 1195   1197# 1197   1199# 1199   1201# 1201
1203# 1203   1205# 1205   1207# 1207   1209# 1209   1211# 1211   1213# 1213   1215# 1215
1217# 1217   1219# 1219   1221# 1221   1223# 1223   1225# 1225   1227# 1227   1229# 1229
1231# 1231   1233# 1233   1235# 1235   1237# 1237   1239# 1239   1241# 1241   1243# 1243
1245# 1245   1247# 1247   1249# 1249   1251# 1251   1253# 1253   1255# 1255   1257# 1257
1259# 1259   1261# 1261   1263# 1263   1265# 1265   1267# 1267   1269# 1269   1271# 1271
```

```
1273# 1273   1275# 1275   1277# 1277   1279# 1279   1281# 1281   1283# 1283   1285# 1285
```

```
1273#  1273   1275#  1275   1277#  1277   1279#  1279   1281#  1281   1283#  1283   1285#  1285     SEQ 1147
1287#  1287   1289#  1289   1291#  1291   1293#  1293   1295#  1295   1297#  1297   1299#  1299
1301#  1301   1303#  1303   1305#  1305   1307#  1307   1309#  1309   1311#  1311   1313#  1313
1315#  1315   1317#  1317   1319#  1319   1321#  1321   1323#  1323   1325#  1325   1327#  1327
1329#  1329   1331#  1331   1333#  1333   1335#  1335   1337#  1337   1339#  1339   1341#  1341
1343#  1343   1345#  1345   1347#  1347   1349#  1349   1351#  1351   1353#  1353   1355#  1355
1357#  1357   1359#  1359   1361#  1361   1363#  1363   1365#  1365   1367#  1367   1369#  1369
1371#  1371   1373#  1373   1375#  1375   1377#  1377   1379#  1379   1381#  1381   1383#  1383
1385#  1385   1387#  1387   1389#  1389   1391#  1391   1393#  1393   1395#  1395   1397#  1397
1399#  1399   1401#  1401   1403#  1403   1405#  1405   1407#  1407   1409#  1409   1411#  1411
1413#  1413   1415#  1415   1417#  1417   1419#  1419   1421#  1421   1423#  1423   1425#  1425
1427#  1427   1429#  1429   1431#  1431   1433#  1433   1435#  1435   1437#  1437   1439#  1439
1441#  1441   1443#  1443   1445#  1445   1447#  1447   1449#  1449   1451#  1451   1453#  1453
1455#  1455   1457#  1457   1459#  1459   1461#  1461   1463#  1463   1465#  1465   1467#  1467
1469#  1469   1471#  1471   1473#  1473   1475#  1475   1477#  1477   1479#  1479   1481#  1481
1483#  1483   1485#  1485   1487#  1487   1489#  1489   1491#  1491   1493#  1493   1495#  1495
1497#  1497   1499#  1499   1501#  1501   1503#  1503   1505#  1505   1507#  1507   1509#  1509
1511#  1511   1513#  1513   1515#  1515   1517#  1517   1519#  1519   1521#  1521   1523#  1523
1525#  1525   1527#  1527   1529#  1529   1531#  1531   1533#  1533   1535#  1535   1537#  1537
1539#  1539   1541#  1541   1543#  1543   1545#  1545   1547#  1547   1549#  1549   1551#  1551
1553#  1553   1555#  1555   1557#  1557   1559#  1559   1561#  1561   1563#  1563   1565#  1565
1567#  1567   1569#  1569   1571#  1571   1573#  1573   1575#  1575   1577#  1577   1579#  1579
1581#  1581   1583#  1583   1585#  1585   1587#  1587   1589#  1589   1591#  1591   1593#  1593
1595#  1595   1597#  1597   1599#  1599   1601#  1601   1603#  1603   1605#  1605   1607#  1607
1609#  1609   1611#  1611   1613#  1613   1615#  1615   1617#  1617   1619#  1619   1621#  1621
1623#  1623   1625#  1625   1627#  1627   1629#  1629   1631#  1631   1633#  1633   1635#  1635
1637#  1637   1639#  1639   1641#  1641   1643#  1643   1645#  1645   1647#  1647   1649#  1649
1651#  1651   1653#  1653   1655#  1655   1657#  1657   1659#  1659   1661#  1661   1663#  1663
1665#  1665   1667#  1667   1669#  1669   1671#  1671   1673#  1673   1675#  1675   1677#  1677
1679#  1679   1681#  1681   1683#  1683   1685#  1685   1687#  1687   1689#  1689   1691#  1691
1693#  1693   1695#  1695   1697#  1697   1699#  1699   1701#  1701   1703#  1703   1705#  1705
1707#  1707   1709#  1709   1711#  1711   1713#  1713   1715#  1715   1717#  1717   1719#  1719
1721#  1721   1723#  1723   1725#  1725   1727#  1727   1729#  1729   1731#  1731   1733#  1733
1735#  1735   1737#  1737   1739#  1739   1741#  1741   1743#  1743   1745#  1745   1747#  1747
1749#  1749   1751#  1751   1753#  1753   1755#  1755   1757#  1757   1759#  1759   1761#  1761
1763#  1763   1765#  1765   1767#  1767   1769#  1769   1771#  1771   1773#  1773   1775#  1775
1777#  1777   1779#  1779   1781#  1781   1783#  1783   1785#  1785   1787#  1787   1789#  1789
1791#  1791   1793#  1793   1795#  1795   1797#  1797   1799#  1799   1801#  1801   1803#  1803
1805#  1805   1807#  1807   1809#  1809   1811#  1811   1813#  1813   1815#  1815   1817#  1817
1819#  1819   1821#  1821   1823#  1823   1825#  1825   1827#  1827   1829#  1829   1831#  1831
1833#  1833   1835#  1835   1837#  1837   1839#  1839   1841#  1841   1843#  1843   1845#  1845
1847#  1847   1849#  1849   1851#  1851   1853#  1853   1855#  1855   1857#  1857   1859#  1859
1861#  1861   1863#  1863   1865#  1865   1867#  1867   1869#  1869   1871#  1871   1873#  1873
1875#  1875   1877#  1877   1879#  1879   1881#  1881   1883#  1883   1885#  1885   1887#  1887
1889#  1889   1891#  1891   1893#  1893   1895#  1895   1897#  1897   1899#  1899   1901#  1901
1903#  1903   1905#  1905   1907#  1907   1909#  1909   1911#  1911   1913#  1913   1915#  1915
1917#  1917   1919#  1919   1921#  1921   1923#  1923   1925#  1925   1927#  1927   1929#  1929
1931#  1931   1933#  1933   1935#  1935   1937#  1937   1939#  1939   1941#  1941   1943#  1943
1945#  1945   1947#  1947   1949#  1949   1951#  1951   1953#  1953   1955#  1955   1957#  1957
1959#  1959   1961#  1961   1963#  1963   1965#  1965   1967#  1967   1969#  1969   1971#  1971
1973#  1973   1975#  1975   1977#  1977   1979#  1979   1981#  1981   1983#  1983   1985#  1985
1987#  1987   1989#  1989   1991#  1991   1993#  1993   1995#  1995   1997#  1997   1999#  1999
2001#  2001   2003#  2003   2005#  2005   2007#  2007   2009#  2009   2011#  2011   2013#  2013
```

```
        2015#   2015    2017#   2017    2019#   2019    2021#   2021    2023#   2023    2025#   2025    2027#   2027      SEQ 1148
        2029#   2029    2031#   2031    2033#   2033    2035#   2035    2037#   2037    2039#   2039    2041#   2041
        2043#   2043    2045#   2045    2047#   2047    2049#   2049    2051#   2051    2053#   2053    2055#   2055
        2057#   2057    2059#   2059    2061#   2061    2063#   2063    2065#   2065    2067#   2067    2069#   2069
        2071#   2071    2073#   2073    2075#   2075    2077#   2077    2082#   2082    2087#   2087
%ADDR   116#    116     121     121#    123     123#    125     125#    127     127#    129     129#    131     131#
        133     133#    135     135#    137     137#    139     139#    141     141#    143     143#    145     145#
        147     147#    149     149#    151     151#    153     153#    161#    161     163     163#    165     165#
        167     167#    169     169#    171     171#    173     173#    175     175#    177     177#    179     179#
        181     181#    183     183#    185     185#    187     187#    189     189#    191     191#    193     193#
        195     195#    197     197#    199     199#    201     201#    203     203#    205     205#    207     207#
        209     209#    211     211#    213     213#    215     215#    217     217#    219     219#    221     221#
        223     223#    225     225#    227     227#    229     229#    231     231#    233     233#    235     235#
        237     237#    239     239#    241     241#    243     243#    245     245#    247     247#    249     249#
        251     251#    253     253#    255     255#    257     257#    259     259#    261     261#    263     263#
        265     265#    267     267#    269     269#    271     271#    273     273#    275     275#    277     277#
        279     279#    281     281#    283     283#    285     285#    287     287#    289     289#    291     291#
        293     293#    295     295#    297     297#    299     299#    301     301#    303     303#    305     305#
        307     307#    309     309#    311     311#    313     313#    315     315#    317     317#    319     319#
        321     321#    323     323#    325     325#    327     327#    329     329#    331     331#    333     333#
        335     335#    337     337#    339     339#    341     341#    343     343#    345     345#    347     347#
        349     349#    351     351#    353     353#    355     355#    357     357#    359     359#    361     361#
        363     363#    365     365#    367     367#    369     369#    371     371#    373     373#    375     375#
        377     377#    379     379#    381     381#    383     383#    385     385#    387     387#    389     389#
        391     391#    393     393#    395     395#    397     397#    399     399#    401     401#    403     403#
        405     405#    407     407#    409     409#    411     411#    413     413#    415     415#    417     417#
        419     419#    421     421#    423     423#    425     425#    427     427#    429     429#    431     431#
        433     433#    435     435#    437     437#    439     439#    441     441#    443     443#    445     445#
        447     447#    449     449#    451     451#    453     453#    455     455#    457     457#    459     459#
        461     461#    463     463#    465     465#    467     467#    469     469#    471     471#    473     473#
        475     475#    477     477#    479     479#    481     481#    483     483#    485     485#    487     487#
        489     489#    491     491#    493     493#    495     495#    497     497#    499     499#    501     501#
        503     503#    505     505#    507     507#    509     509#    511     511#    513     513#    515     515#
        517     517#    519     519#    521     521#    523     523#    525     525#    527     527#    529     529#
        531     531#    533     533#    535     535#    537     537#    539     539#    541     541#    543     543#
        545     545#    547     547#    549     549#    551     551#    553     553#    555     555#    557     557#
        559     559#    561     561#    563     563#    565     565#    567     567#    569     569#    571     571#
        573     573#    575     575#    577     577#    579     579#    581     581#    583     583#    585     585#
        587     587#    589     589#    591     591#    593     593#    595     595#    597     597#    599     599#
        601     601#    603     603#    605     605#    607     607#    609     609#    611     611#    613     613#
        615     615#    617     617#    619     619#    621     621#    623     623#    625     625#    627     627#
        629     629#    631     631#    633     633#    635     635#    637     637#    639     639#    641     641#
        643     643#    645     645#    647     647#    649     649#    651     651#    653     653#    655     655#
        657     657#    659     659#    661     661#    663     663#    665     665#    667     667#    669     669#
        671     671#    673     673#    675     675#    677     677#    679     679#    681     681#    683     683#
        685     685#    687     687#    689     689#    691     691#    693     693#    695     695#    697     697#
        699     699#    701     701#    703     703#    705     705#    707     707#    709     709#    711     711#
        713     713#    715     715#    717     717#    719     719#    721     721#    723     723#    725     725#
        727     727#    729     729#    731     731#    733     733#    735     735#    737     737#    739     739#
        741     741#    743     743#    745     745#    747     747#    749     749#    751     751#    753     753#
        755     755#    757     757#    759     759#    761     761#    763     763#    765     765#    767     767#
        769     769#    771     771#    773     773#    775     775#    777     777#    779     779#    781     781#
        783     783#    785     785#    787     787#    789     789#    791     791#    793     793#    795     795#
```

```
 797    797#   799    799#   801    801#   803    803#   805    805#   807    807#   809    809#
 811    811#   813    813#   815    815#   817    817#   819    819#   821    821#   823    823#
 825    825#   827    827#   829    829#   831    831#   833    833#   835    835#   837    837#
 839    839#   841    841#   843    843#   845    845#   847    847#   849    849#   851    851#
 853    853#   855    855#   857    857#   859    859#   861    861#   863    863#   865    865#
 867    867#   869    869#   871    871#   873    873#   875    875#   877    877#   879    879#
 881    881#   883    883#   885    885#   887    887#   889    889#   891    891#   893    893#
 895    895#   897    897#   899    899#   901    901#   903    903#   905    905#   907    907#
 909    909#   911    911#   913    913#   915    915#   917    917#   919    919#   921    921#
 923    923#   925    925#   927    927#   929    929#   931    931#   933    933#   935    935#
 937    937#   939    939#   941    941#   943    943#   945    945#   947    947#   949    949#
 951    951#   953    953#   955    955#   957    957#   959    959#   961    961#   963    963#
 965    965#   967    967#   969    969#   971    971#   973    973#   975    975#   977    977#
 979    979#   981    981#   983    983#   985    985#   987    987#   989    989#   991    991#
 993    993#   995    995#   997    997#   999    999#  1001   1001#  1003   1003#  1005   1005#
1007   1007#  1009   1009#  1011   1011#  1013   1013#  1015   1015#  1017   1017#  1019   1019#
1021   1021#  1023   1023#  1025   1025#  1027   1027#  1029   1029#  1031   1031#  1033   1033#
1035   1035#  1037   1037#  1039   1039#  1041   1041#  1043   1043#  1045   1045#  1047   1047#
1049   1049#  1051   1051#  1053   1053#  1055   1055#  1057   1057#  1059   1059#  1061   1061#
1063   1063#  1065   1065#  1067   1067#  1069   1069#  1071   1071#  1073   1073#  1075   1075#
1077   1077#  1079   1079#  1081   1081#  1083   1083#  1085   1085#  1087   1087#  1089   1089#
1091   1091#  1093   1093#  1095   1095#  1097   1097#  1099   1099#  1101   1101#  1103   1103#
1105   1105#  1107   1107#  1109   1109#  1111   1111#  1113   1113#  1115   1115#  1117   1117#
1119   1119#  1121   1121#  1123   1123#  1125   1125#  1127   1127#  1129   1129#  1131   1131#
1133   1133#  1135   1135#  1137   1137#  1139   1139#  1141   1141#  1143   1143#  1145   1145#
1147   1147#  1149   1149#  1151   1151#  1153   1153#  1155   1155#  1157   1157#  1159   1159#
1161   1161#  1163   1163#  1165   1165#  1167   1167#  1169   1169#  1171   1171#  1173   1173#
1175   1175#  1177   1177#  1179   1179#  1181   1181#  1183   1183#  1185   1185#  1187   1187#
1189   1189#  1191   1191#  1193   1193#  1195   1195#  1197   1197#  1199   1199#  1201   1201#
1203   1203#  1205   1205#  1207   1207#  1209   1209#  1211   1211#  1213   1213#  1215   1215#
1217   1217#  1219   1219#  1221   1221#  1223   1223#  1225   1225#  1227   1227#  1229   1229#
1231   1231#  1233   1233#  1235   1235#  1237   1237#  1239   1239#  1241   1241#  1243   1243#
1245   1245#  1247   1247#  1249   1249#  1251   1251#  1253   1253#  1255   1255#  1257   1257#
1259   1259#  1261   1261#  1263   1263#  1265   1265#  1267   1267#  1269   1269#  1271   1271#
1273   1273#  1275   1275#  1277   1277#  1279   1279#  1281   1281#  1283   1283#  1285   1285#
1287   1287#  1289   1289#  1291   1291#  1293   1293#  1295   1295#  1297   1297#  1299   1299#
1301   1301#  1303   1303#  1305   1305#  1307   1307#  1309   1309#  1311   1311#  1313   1313#
1315   1315#  1317   1317#  1319   1319#  1321   1321#  1323   1323#  1325   1325#  1327   1327#
1329   1329#  1331   1331#  1333   1333#  1335   1335#  1337   1337#  1339   1339#  1341   1341#
1343   1343#  1345   1345#  1347   1347#  1349   1349#  1351   1351#  1353   1353#  1355   1355#
1357   1357#  1359   1359#  1361   1361#  1363   1363#  1365   1365#  1367   1367#  1369   1369#
1371   1371#  1373   1373#  1375   1375#  1377   1377#  1379   1379#  1381   1381#  1383   1383#
1385   1385#  1387   1387#  1389   1389#  1391   1391#  1393   1393#  1395   1395#  1397   1397#
1399   1399#  1401   1401#  1403   1403#  1405   1405#  1407   1407#  1409   1409#  1411   1411#
1413   1413#  1415   1415#  1417   1417#  1419   1419#  1421   1421#  1423   1423#  1425   1425#
1427   1427#  1429   1429#  1431   1431#  1433   1433#  1435   1435#  1437   1437#  1439   1439#
1441   1441#  1443   1443#  1445   1445#  1447   1447#  1449   1449#  1451   1451#  1453   1453#
1455   1455#  1457   1457#  1459   1459#  1461   1461#  1463   1463#  1465   1465#  1467   1467#
1469   1469#  1471   1471#  1473   1473#  1475   1475#  1477   1477#  1479   1479#  1481   1481#
1483   1483#  1485   1485#  1487   1487#  1489   1489#  1491   1491#  1493   1493#  1495   1495#
1497   1497#  1499   1499#  1501   1501#  1503   1503#  1505   1505#  1507   1507#  1509   1509#
1511   1511#  1513   1513#  1515   1515#  1517   1517#  1519   1519#  1521   1521#  1523   1523#
1525   1525#  1527   1527#  1529   1529#  1531   1531#  1533   1533#  1535   1535#  1537   1537#
```

```
        1539    1539#   1541    1541#   1543    1543#   1545    1545#   1547    1547#   1549    1549#   1551    1551#      SEQ 1150
        1553    1553#   1555    1555#   1557    1557#   1559    1559#   1561    1561#   1563    1563#   1565    1565#
        1567    1567#   1569    1569#   1571    1571#   1573    1573#   1575    1575#   1577    1577#   1579    1579#
        1581    1581#   1583    1583#   1585    1585#   1587    1587#   1589    1589#   1591    1591#   1593    1593#
        1595    1595#   1597    1597#   1599    1599#   1601    1601#   1603    1603#   1605    1605#   1607    1607#
        1609    1609#   1611    1611#   1613    1613#   1615    1615#   1617    1617#   1619    1619#   1621    1621#
        1623    1623#   1625    1625#   1627    1627#   1629    1629#   1631    1631#   1633    1633#   1635    1635#
        1637    1637#   1639    1639#   1641    1641#   1643    1643#   1645    1645#   1647    1647#   1649    1649#
        1651    1651#   1653    1653#   1655    1655#   1657    1657#   1659    1659#   1661    1661#   1663    1663#
        1665    1665#   1667    1667#   1669    1669#   1671    1671#   1673    1673#   1675    1675#   1677    1677#
        1679    1679#   1681    1681#   1683    1683#   1685    1685#   1687    1687#   1689    1689#   1691    1691#
        1693    1693#   1695    1695#   1697    1697#   1699    1699#   1701    1701#   1703    1703#   1705    1705#
        1707    1707#   1709    1709#   1711    1711#   1713    1713#   1715    1715#   1717    1717#   1719    1719#
        1721    1721#   1723    1723#   1725    1725#   1727    1727#   1729    1729#   1731    1731#   1733    1733#
        1735    1735#   1737    1737#   1739    1739#   1741    1741#   1743    1743#   1745    1745#   1747    1747#
        1749    1749#   1751    1751#   1753    1753#   1755    1755#   1757    1757#   1759    1759#   1761    1761#
        1763    1763#   1765    1765#   1767    1767#   1769    1769#   1771    1771#   1773    1773#   1775    1775#
        1777    1777#   1779    1779#   1781    1781#   1783    1783#   1785    1785#   1787    1787#   1789    1789#
        1791    1791#   1793    1793#   1795    1795#   1797    1797#   1799    1799#   1801    1801#   1803    1803#
        1805    1805#   1807    1807#   1809    1809#   1811    1811#   1813    1813#   1815    1815#   1817    1817#
        1819    1819#   1821    1821#   1823    1823#   1825    1825#   1827    1827#   1829    1829#   1831    1831#
        1833    1833#   1835    1835#   1837    1837#   1839    1839#   1841    1841#   1843    1843#   1845    1845#
        1847    1847#   1849    1849#   1851    1851#   1853    1853#   1855    1855#   1857    1857#   1859    1859#
        1861    1861#   1863    1863#   1865    1865#   1867    1867#   1869    1869#   1871    1871#   1873    1873#
        1875    1875#   1877    1877#   1879    1879#   1881    1881#   1883    1883#   1885    1885#   1887    1887#
        1889    1889#   1891    1891#   1893    1893#   1895    1895#   1897    1897#   1899    1899#   1901    1901#
        1903    1903#   1905    1905#   1907    1907#   1909    1909#   1911    1911#   1913    1913#   1915    1915#
        1917    1917#   1919    1919#   1921    1921#   1923    1923#   1925    1925#   1927    1927#   1929    1929#
        1931    1931#   1933    1933#   1935    1935#   1937    1937#   1939    1939#   1941    1941#   1943    1943#
        1945    1945#   1947    1947#   1949    1949#   1951    1951#   1953    1953#   1955    1955#   1957    1957#
        1959    1959#   1961    1961#   1963    1963#   1965    1965#   1967    1967#   1969    1969#   1971    1971#
        1973    1973#   1975    1975#   1977    1977#   1979    1979#   1981    1981#   1983    1983#   1985    1985#
        1987    1987#   1989    1989#   1991    1991#   1993    1993#   1995    1995#   1997    1997#   1999    1999#
        2001    2001#   2003    2003#   2005    2005#   2007    2007#   2009    2009#   2011    2011#   2013    2013#
        2015    2015#   2017    2017#   2019    2019#   2021    2021#   2023    2023#   2025    2025#   2027    2027#
        2029    2029#   2031    2031#   2033    2033#   2035    2035#   2037    2037#   2039    2039#   2041    2041#
        2043    2043#   2045    2045#   2047    2047#   2049    2049#   2051    2051#   2053    2053#   2055    2055#
        2057    2057#   2059    2059#   2061    2061#   2063    2063#   2065    2065#   2067    2067#   2069    2069#
        2071    2071#   2073    2073#   2075    2075#   2077    2077#   2082    2082#   2087#   2087
%ML      116#    116     121#    121     123#    123     125#    125     127#    127     129#    129     131#    131
         133#    133     135#    135     137#    137     139#    139     141#    141     143#    143     145#    145
         147#    147     149#    149     151#    151     153#    153     161#    161     163#    163     165#    165
         167#    167     169#    169     171#    171     173#    173     175#    175     177#    177     179#    179
         181#    181     183#    183     185#    185     187#    187     189#    189     191#    191     193#    193
         195#    195     197#    197     199#    199     201#    201     203#    203     205#    205     207#    207
         209#    209     211#    211     213#    213     215#    215     217#    217     219#    219     221#    221
         223#    223     225#    225     227#    227     229#    229     231#    231     233#    233     235#    235
         237#    237     239#    239     241#    241     243#    243     245#    245     247#    247     249#    249
         251#    251     253#    253     255#    255     257#    257     259#    259     261#    261     263#    263
         265#    265     267#    267     269#    269     271#    271     273#    273     275#    275     277#    277
         279#    279     281#    281     283#    283     285#    285     287#    287     289#    289     291#    291
         293#    293     295#    295     297#    297     299#    299     301#    301     303#    303     305#    305
         307#    307     309#    309     311#    311     313#    313     315#    315     317#    317     319#    319
```

```
 321#  321   323#  323   325#  325   327#  327   329#  329   331#  331   333#  333     SEQ 1151
 335#  335   337#  337   339#  339   341#  341   343#  343   345#  345   347#  347
 349#  349   351#  351   353#  353   355#  355   357#  357   359#  359   361#  361
 363#  363   365#  365   367#  367   369#  369   371#  371   373#  373   375#  375
 377#  377   379#  379   381#  381   383#  383   385#  385   387#  387   389#  389
 391#  391   393#  393   395#  395   397#  397   399#  399   401#  401   403#  403
 405#  405   407#  407   409#  409   411#  411   413#  413   415#  415   417#  417
 419#  419   421#  421   423#  423   425#  425   427#  427   429#  429   431#  431
 433#  433   435#  435   437#  437   439#  439   441#  441   443#  443   445#  445
 447#  447   449#  449   451#  451   453#  453   455#  455   457#  457   459#  459
 461#  461   463#  463   465#  465   467#  467   469#  469   471#  471   473#  473
 475#  475   477#  477   479#  479   481#  481   483#  483   485#  485   487#  487
 489#  489   491#  491   493#  493   495#  495   497#  497   499#  499   501#  501
 503#  503   505#  505   507#  507   509#  509   511#  511   513#  513   515#  515
 517#  517   519#  519   521#  521   523#  523   525#  525   527#  527   529#  529
 531#  531   533#  533   535#  535   537#  537   539#  539   541#  541   543#  543
 545#  545   547#  547   549#  549   551#  551   553#  553   555#  555   557#  557
 559#  559   561#  561   563#  563   565#  565   567#  567   569#  569   571#  571
 573#  573   575#  575   577#  577   579#  579   581#  581   583#  583   585#  585
 587#  587   589#  589   591#  591   593#  593   595#  595   597#  597   599#  599
 601#  601   603#  603   605#  605   607#  607   609#  609   611#  611   613#  613
 615#  615   617#  617   619#  619   621#  621   623#  623   625#  625   627#  627
 629#  629   631#  631   633#  633   635#  635   637#  637   639#  639   641#  641
 643#  643   645#  645   647#  647   649#  649   651#  651   653#  653   655#  655
 657#  657   659#  659   661#  661   663#  663   665#  665   667#  667   669#  669
 671#  671   673#  673   675#  675   677#  677   679#  679   681#  681   683#  683
 685#  685   687#  687   689#  689   691#  691   693#  693   695#  695   697#  697
 699#  699   701#  701   703#  703   705#  705   707#  707   709#  709   711#  711
 713#  713   715#  715   717#  717   719#  719   721#  721   723#  723   725#  725
 727#  727   729#  729   731#  731   733#  733   735#  735   737#  737   739#  739
 741#  741   743#  743   745#  745   747#  747   749#  749   751#  751   753#  753
 755#  755   757#  757   759#  759   761#  761   763#  763   765#  765   767#  767
 769#  769   771#  771   773#  773   775#  775   777#  777   779#  779   781#  781
 783#  783   785#  785   787#  787   789#  789   791#  791   793#  793   795#  795
 797#  797   799#  799   801#  801   803#  803   805#  805   807#  807   809#  809
 811#  811   813#  813   815#  815   817#  817   819#  819   821#  821   823#  823
 825#  825   827#  827   829#  829   831#  831   833#  833   835#  835   837#  837
 839#  839   841#  841   843#  843   845#  845   847#  847   849#  849   851#  851
 853#  853   855#  855   857#  857   859#  859   861#  861   863#  863   865#  865
 867#  867   869#  869   871#  871   873#  873   875#  875   877#  877   879#  879
 881#  881   883#  883   885#  885   887#  887   889#  889   891#  891   893#  893
 895#  895   897#  897   899#  899   901#  901   903#  903   905#  905   907#  907
 909#  909   911#  911   913#  913   915#  915   917#  917   919#  919   921#  921
 923#  923   925#  925   927#  927   929#  929   931#  931   933#  933   935#  935
 937#  937   939#  939   941#  941   943#  943   945#  945   947#  947   949#  949
 951#  951   953#  953   955#  955   957#  957   959#  959   961#  961   963#  963
 965#  965   967#  967   969#  969   971#  971   973#  973   975#  975   977#  977
 979#  979   981#  981   983#  983   985#  985   987#  987   989#  989   991#  991
 993#  993   995#  995   997#  997   999#  999  1001# 1001  1003# 1003  1005# 1005
1007# 1007  1009# 1009  1011# 1011  1013# 1013  1015# 1015  1017# 1017  1019# 1019
1021# 1021  1023# 1023  1025# 1025  1027# 1027  1029# 1029  1031# 1031  1033# 1033
1035# 1035  1037# 1037  1039# 1039  1041# 1041  1043# 1043  1045# 1045  1047# 1047
1049# 1049  1051# 1051  1053# 1053  1055# 1055  1057# 1057  1059# 1059  1061# 1061
```

```
1063#  1063   1065#  1065   1067#  1067   1069#  1069   1071#  1071   1073#  1073   1075#  1075
1077#  1077   1079#  1079   1081#  1081   1083#  1083   1085#  1085   1087#  1087   1089#  1089
1091#  1091   1093#  1093   1095#  1095   1097#  1097   1099#  1099   1101#  1101   1103#  1103
1105#  1105   1107#  1107   1109#  1109   1111#  1111   1113#  1113   1115#  1115   1117#  1117
1119#  1119   1121#  1121   1123#  1123   1125#  1125   1127#  1127   1129#  1129   1131#  1131
1133#  1133   1135#  1135   1137#  1137   1139#  1139   1141#  1141   1143#  1143   1145#  1145
1147#  1147   1149#  1149   1151#  1151   1153#  1153   1155#  1155   1157#  1157   1159#  1159
1161#  1161   1163#  1163   1165#  1165   1167#  1167   1169#  1169   1171#  1171   1173#  1173
1175#  1175   1177#  1177   1179#  1179   1181#  1181   1183#  1183   1185#  1185   1187#  1187
1189#  1189   1191#  1191   1193#  1193   1195#  1195   1197#  1197   1199#  1199   1201#  1201
1203#  1203   1205#  1205   1207#  1207   1209#  1209   1211#  1211   1213#  1213   1215#  1215
1217#  1217   1219#  1219   1221#  1221   1223#  1223   1225#  1225   1227#  1227   1229#  1229
1231#  1231   1233#  1233   1235#  1235   1237#  1237   1239#  1239   1241#  1241   1243#  1243
1245#  1245   1247#  1247   1249#  1249   1251#  1251   1253#  1253   1255#  1255   1257#  1257
1259#  1259   1261#  1261   1263#  1263   1265#  1265   1267#  1267   1269#  1269   1271#  1271
1273#  1273   1275#  1275   1277#  1277   1279#  1279   1281#  1281   1283#  1283   1285#  1285
1287#  1287   1289#  1289   1291#  1291   1293#  1293   1295#  1295   1297#  1297   1299#  1299
1301#  1301   1303#  1303   1305#  1305   1307#  1307   1309#  1309   1311#  1311   1313#  1313
1315#  1315   1317#  1317   1319#  1319   1321#  1321   1323#  1323   1325#  1325   1327#  1327
1329#  1329   1331#  1331   1333#  1333   1335#  1335   1337#  1337   1339#  1339   1341#  1341
1343#  1343   1345#  1345   1347#  1347   1349#  1349   1351#  1351   1353#  1353   1355#  1355
1357#  1357   1359#  1359   1361#  1361   1363#  1363   1365#  1365   1367#  1367   1369#  1369
1371#  1371   1373#  1373   1375#  1375   1377#  1377   1379#  1379   1381#  1381   1383#  1383
1385#  1385   1387#  1387   1389#  1389   1391#  1391   1393#  1393   1395#  1395   1397#  1397
1399#  1399   1401#  1401   1403#  1403   1405#  1405   1407#  1407   1409#  1409   1411#  1411
1413#  1413   1415#  1415   1417#  1417   1419#  1419   1421#  1421   1423#  1423   1425#  1425
1427#  1427   1429#  1429   1431#  1431   1433#  1433   1435#  1435   1437#  1437   1439#  1439
1441#  1441   1443#  1443   1445#  1445   1447#  1447   1449#  1449   1451#  1451   1453#  1453
1455#  1455   1457#  1457   1459#  1459   1461#  1461   1463#  1463   1465#  1465   1467#  1467
1469#  1469   1471#  1471   1473#  1473   1475#  1475   1477#  1477   1479#  1479   1481#  1481
1483#  1483   1485#  1485   1487#  1487   1489#  1489   1491#  1491   1493#  1493   1495#  1495
1497#  1497   1499#  1499   1501#  1501   1503#  1503   1505#  1505   1507#  1507   1509#  1509
1511#  1511   1513#  1513   1515#  1515   1517#  1517   1519#  1519   1521#  1521   1523#  1523
1525#  1525   1527#  1527   1529#  1529   1531#  1531   1533#  1533   1535#  1535   1537#  1537
1539#  1539   1541#  1541   1543#  1543   1545#  1545   1547#  1547   1549#  1549   1551#  1551
1553#  1553   1555#  1555   1557#  1557   1559#  1559   1561#  1561   1563#  1563   1565#  1565
1567#  1567   1569#  1569   1571#  1571   1573#  1573   1575#  1575   1577#  1577   1579#  1579
1581#  1581   1583#  1583   1585#  1585   1587#  1587   1589#  1589   1591#  1591   1593#  1593
1595#  1595   1597#  1597   1599#  1599   1601#  1601   1603#  1603   1605#  1605   1607#  1607
1609#  1609   1611#  1611   1613#  1613   1615#  1615   1617#  1617   1619#  1619   1621#  1621
1623#  1623   1625#  1625   1627#  1627   1629#  1629   1631#  1631   1633#  1633   1635#  1635
1637#  1637   1639#  1639   1641#  1641   1643#  1643   1645#  1645   1647#  1647   1649#  1649
1651#  1651   1653#  1653   1655#  1655   1657#  1657   1659#  1659   1661#  1661   1663#  1663
1665#  1665   1667#  1667   1669#  1669   1671#  1671   1673#  1673   1675#  1675   1677#  1677
1679#  1679   1681#  1681   1683#  1683   1685#  1685   1687#  1687   1689#  1689   1691#  1691
1693#  1693   1695#  1695   1697#  1697   1699#  1699   1701#  1701   1703#  1703   1705#  1705
1707#  1707   1709#  1709   1711#  1711   1713#  1713   1715#  1715   1717#  1717   1719#  1719
1721#  1721   1723#  1723   1725#  1725   1727#  1727   1729#  1729   1731#  1731   1733#  1733
1735#  1735   1737#  1737   1739#  1739   1741#  1741   1743#  1743   1745#  1745   1747#  1747
1749#  1749   1751#  1751   1753#  1753   1755#  1755   1757#  1757   1759#  1759   1761#  1761
1763#  1763   1765#  1765   1767#  1767   1769#  1769   1771#  1771   1773#  1773   1775#  1775
1777#  1777   1779#  1779   1781#  1781   1783#  1783   1785#  1785   1787#  1787   1789#  1789
1791#  1791   1793#  1793   1795#  1795   1797#  1797   1799#  1799   1801#  1801   1803#  1803
```

```
1140       1805#  1805    1807#  1807    1809#  1809    1811#  1811    1813#  1813    1815#  1815    1817#  1817     SEQ 1153
           1819#  1819    1821#  1821    1823#  1823    1825#  1825    1827#  1827    1829#  1829    1831#  1831
           1833#  1833    1835#  1835    1837#  1837    1839#  1839    1841#  1841    1843#  1843    1845#  1845
           1847#  1847    1849#  1849    1851#  1851    1853#  1853    1855#  1855    1857#  1857    1859#  1859
           1861#  1861    1863#  1863    1865#  1865    1867#  1867    1869#  1869    1871#  1871    1873#  1873
           1875#  1875    1877#  1877    1879#  1879    1881#  1881    1883#  1883    1885#  1885    1887#  1887
           1889#  1889    1891#  1891    1893#  1893    1895#  1895    1897#  1897    1899#  1899    1901#  1901
           1903#  1903    1905#  1905    1907#  1907    1909#  1909    1911#  1911    1913#  1913    1915#  1915
           1917#  1917    1919#  1919    1921#  1921    1923#  1923    1925#  1925    1927#  1927    1929#  1929
           1931#  1931    1933#  1933    1935#  1935    1937#  1937    1939#  1939    1941#  1941    1943#  1943
           1945#  1945    1947#  1947    1949#  1949    1951#  1951    1953#  1953    1955#  1955    1957#  1957
           1959#  1959    1961#  1961    1963#  1963    1965#  1965    1967#  1967    1969#  1969    1971#  1971
           1973#  1973    1975#  1975    1977#  1977    1979#  1979    1981#  1981    1983#  1983    1985#  1985
           1987#  1987    1989#  1989    1991#  1991    1993#  1993    1995#  1995    1997#  1997    1999#  1999
           2001#  2001    2003#  2003    2005#  2005    2007#  2007    2009#  2009    2011#  2011    2013#  2013
           2015#  2015    2017#  2017    2019#  2019    2021#  2021    2023#  2023    2025#  2025    2027#  2027
           2029#  2029    2031#  2031    2033#  2033    2035#  2035    2037#  2037    2039#  2039    2041#  2041
           2043#  2043    2045#  2045    2047#  2047    2049#  2049    2051#  2051    2053#  2053    2055#  2055
           2057#  2057    2059#  2059    2061#  2061    2063#  2063    2065#  2065    2067#  2067    2069#  2069
           2071#  2071    2073#  2073    2075#  2075    2077#  2077    2082#  2082    2087#  2087
%MR         116#   116     121#   121     123#   123     125#   125     127#   127     129#   129     131#   131
            133#   133     135#   135     137#   137     139#   139     141#   141     143#   143     145#   145
            147#   147     149#   149     151#   151     153#   153     161#   161     163#   163     165#   165
            167#   167     169#   169     171#   171     173#   173     175#   175     177#   177     179#   179
            181#   181     183#   183     185#   185     187#   187     189#   189     191#   191     193#   193
            195#   195     197#   197     199#   199     201#   201     203#   203     205#   205     207#   207
            209#   209     211#   211     213#   213     215#   215     217#   217     219#   219     221#   221
            223#   223     225#   225     227#   227     229#   229     231#   231     233#   233     235#   235
            237#   237     239#   239     241#   241     243#   243     245#   245     247#   247     249#   249
            251#   251     253#   253     255#   255     257#   257     259#   259     261#   261     263#   263
            265#   265     267#   267     269#   269     271#   271     273#   273     275#   275     277#   277
            279#   279     281#   281     283#   283     285#   285     287#   287     289#   289     291#   291
            293#   293     295#   295     297#   297     299#   299     301#   301     303#   303     305#   305
            307#   307     309#   309     311#   311     313#   313     315#   315     317#   317     319#   319
            321#   321     323#   323     325#   325     327#   327     329#   329     331#   331     333#   333
            335#   335     337#   337     339#   339     341#   341     343#   343     345#   345     347#   347
            349#   349     351#   351     353#   353     355#   355     357#   357     359#   359     361#   361
            363#   363     365#   365     367#   367     369#   369     371#   371     373#   373     375#   375
            377#   377     379#   379     381#   381     383#   383     385#   385     387#   387     389#   389
            391#   391     393#   393     395#   395     397#   397     399#   399     401#   401     403#   403
            405#   405     407#   407     409#   409     411#   411     413#   413     415#   415     417#   417
            419#   419     421#   421     423#   423     425#   425     427#   427     429#   429     431#   431
            433#   433     435#   435     437#   437     439#   439     441#   441     443#   443     445#   445
            447#   447     449#   449     451#   451     453#   453     455#   455     457#   457     459#   459
            461#   461     463#   463     465#   465     467#   467     469#   469     471#   471     473#   473
            475#   475     477#   477     479#   479     481#   481     483#   483     485#   485     487#   487
            489#   489     491#   491     493#   493     495#   495     497#   497     499#   499     501#   501
            503#   503     505#   505     507#   507     509#   509     511#   511     513#   513     515#   515
            517#   517     519#   519     521#   521     523#   523     525#   525     527#   527     529#   529
            531#   531     533#   533     535#   535     537#   537     539#   539     541#   541     543#   543
            545#   545     547#   547     549#   549     551#   551     553#   553     555#   555     557#   557
            559#   559     561#   561     563#   563     565#   565     567#   567     569#   569     571#   571
            573#   573     575#   575     577#   577     579#   579     581#   581     583#   583     585#   585
```

```
 587#   587   589#   589   591#   591   593#   593   595#   595   597#   597   599#   599
 601#   601   603#   603   605#   605   607#   607   609#   609   611#   611   613#   613
 615#   615   617#   617   619#   619   621#   621   623#   623   625#   625   627#   627
 629#   629   631#   631   633#   633   635#   635   637#   637   639#   639   641#   641
 643#   643   645#   645   647#   647   649#   649   651#   651   653#   653   655#   655
 657#   657   659#   659   661#   661   663#   663   665#   665   667#   667   669#   669
 671#   671   673#   673   675#   675   677#   677   679#   679   681#   681   683#   683
 685#   685   687#   687   689#   689   691#   691   693#   693   695#   695   697#   697
 699#   699   701#   701   703#   703   705#   705   707#   707   709#   709   711#   711
 713#   713   715#   715   717#   717   719#   719   721#   721   723#   723   725#   725
 727#   727   729#   729   731#   731   733#   733   735#   735   737#   737   739#   739
 741#   741   743#   743   745#   745   747#   747   749#   749   751#   751   753#   753
 755#   755   757#   757   759#   759   761#   761   763#   763   765#   765   767#   767
 769#   769   771#   771   773#   773   775#   775   777#   777   779#   779   781#   781
 783#   783   785#   785   787#   787   789#   789   791#   791   793#   793   795#   795
 797#   797   799#   799   801#   801   803#   803   805#   805   807#   807   809#   809
 811#   811   813#   813   815#   815   817#   817   819#   819   821#   821   823#   823
 825#   825   827#   827   829#   829   831#   831   833#   833   835#   835   837#   837
 839#   839   841#   841   843#   843   845#   845   847#   847   849#   849   851#   851
 853#   853   855#   855   857#   857   859#   859   861#   861   863#   863   865#   865
 867#   867   869#   869   871#   871   873#   873   875#   875   877#   877   879#   879
 881#   881   883#   883   885#   885   887#   887   889#   889   891#   891   893#   893
 895#   895   897#   897   899#   899   901#   901   903#   903   905#   905   907#   907
 909#   909   911#   911   913#   913   915#   915   917#   917   919#   919   921#   921
 923#   923   925#   925   927#   927   929#   929   931#   931   933#   933   935#   935
 937#   937   939#   939   941#   941   943#   943   945#   945   947#   947   949#   949
 951#   951   953#   953   955#   955   957#   957   959#   959   961#   961   963#   963
 965#   965   967#   967   969#   969   971#   971   973#   973   975#   975   977#   977
 979#   979   981#   981   983#   983   985#   985   987#   987   989#   989   991#   991
 993#   993   995#   995   997#   997   999#   999  1001#  1001  1003#  1003  1005#  1005
1007#  1007  1009#  1009  1011#  1011  1013#  1013  1015#  1015  1017#  1017  1019#  1019
1021#  1021  1023#  1023  1025#  1025  1027#  1027  1029#  1029  1031#  1031  1033#  1033
1035#  1035  1037#  1037  1039#  1039  1041#  1041  1043#  1043  1045#  1045  1047#  1047
1049#  1049  1051#  1051  1053#  1053  1055#  1055  1057#  1057  1059#  1059  1061#  1061
1063#  1063  1065#  1065  1067#  1067  1069#  1069  1071#  1071  1073#  1073  1075#  1075
1077#  1077  1079#  1079  1081#  1081  1083#  1083  1085#  1085  1087#  1087  1089#  1089
1091#  1091  1093#  1093  1095#  1095  1097#  1097  1099#  1099  1101#  1101  1103#  1103
1105#  1105  1107#  1107  1109#  1109  1111#  1111  1113#  1113  1115#  1115  1117#  1117
1119#  1119  1121#  1121  1123#  1123  1125#  1125  1127#  1127  1129#  1129  1131#  1131
1133#  1133  1135#  1135  1137#  1137  1139#  1139  1141#  1141  1143#  1143  1145#  1145
1147#  1147  1149#  1149  1151#  1151  1153#  1153  1155#  1155  1157#  1157  1159#  1159
1161#  1161  1163#  1163  1165#  1165  1167#  1167  1169#  1169  1171#  1171  1173#  1173
1175#  1175  1177#  1177  1179#  1179  1181#  1181  1183#  1183  1185#  1185  1187#  1187
1189#  1189  1191#  1191  1193#  1193  1195#  1195  1197#  1197  1199#  1199  1201#  1201
1203#  1203  1205#  1205  1207#  1207  1209#  1209  1211#  1211  1213#  1213  1215#  1215
1217#  1217  1219#  1219  1221#  1221  1223#  1223  1225#  1225  1227#  1227  1229#  1229
1231#  1231  1233#  1233  1235#  1235  1237#  1237  1239#  1239  1241#  1241  1243#  1243
1245#  1245  1247#  1247  1249#  1249  1251#  1251  1253#  1253  1255#  1255  1257#  1257
1259#  1259  1261#  1261  1263#  1263  1265#  1265  1267#  1267  1269#  1269  1271#  1271
1273#  1273  1275#  1275  1277#  1277  1279#  1279  1281#  1281  1283#  1283  1285#  1285
1287#  1287  1289#  1289  1291#  1291  1293#  1293  1295#  1295  1297#  1297  1299#  1299
1301#  1301  1303#  1303  1305#  1305  1307#  1307  1309#  1309  1311#  1311  1313#  1313
1315#  1315  1317#  1317  1319#  1319  1321#  1321  1323#  1323  1325#  1325  1327#  1327
```

```
1329#  1329   1331#  1331   1333#  1333   1335#  1335   1337#  1337   1339#  1339   1341#  1341
1343#  1343   1345#  1345   1347#  1347   1349#  1349   1351#  1351   1353#  1353   1355#  1355
1357#  1357   1359#  1359   1361#  1361   1363#  1363   1365#  1365   1367#  1367   1369#  1369
1371#  1371   1373#  1373   1375#  1375   1377#  1377   1379#  1379   1381#  1381   1383#  1383
1385#  1385   1387#  1387   1389#  1389   1391#  1391   1393#  1393   1395#  1395   1397#  1397
1399#  1399   1401#  1401   1403#  1403   1405#  1405   1407#  1407   1409#  1409   1411#  1411
1413#  1413   1415#  1415   1417#  1417   1419#  1419   1421#  1421   1423#  1423   1425#  1425
1427#  1427   1429#  1429   1431#  1431   1433#  1433   1435#  1435   1437#  1437   1439#  1439
1441#  1441   1443#  1443   1445#  1445   1447#  1447   1449#  1449   1451#  1451   1453#  1453
1455#  1455   1457#  1457   1459#  1459   1461#  1461   1463#  1463   1465#  1465   1467#  1467
1469#  1469   1471#  1471   1473#  1473   1475#  1475   1477#  1477   1479#  1479   1481#  1481
1483#  1483   1485#  1485   1487#  1487   1489#  1489   1491#  1491   1493#  1493   1495#  1495
1497#  1497   1499#  1499   1501#  1501   1503#  1503   1505#  1505   1507#  1507   1509#  1509
1511#  1511   1513#  1513   1515#  1515   1517#  1517   1519#  1519   1521#  1521   1523#  1523
1525#  1525   1527#  1527   1529#  1529   1531#  1531   1533#  1533   1535#  1535   1537#  1537
1539#  1539   1541#  1541   1543#  1543   1545#  1545   1547#  1547   1549#  1549   1551#  1551
1553#  1553   1555#  1555   1557#  1557   1559#  1559   1561#  1561   1563#  1563   1565#  1565
1567#  1567   1569#  1569   1571#  1571   1573#  1573   1575#  1575   1577#  1577   1579#  1579
1581#  1581   1583#  1583   1585#  1585   1587#  1587   1589#  1589   1591#  1591   1593#  1593
1595#  1595   1597#  1597   1599#  1599   1601#  1601   1603#  1603   1605#  1605   1607#  1607
1609#  1609   1611#  1611   1613#  1613   1615#  1615   1617#  1617   1619#  1619   1621#  1621
1623#  1623   1625#  1625   1627#  1627   1629#  1629   1631#  1631   1633#  1633   1635#  1635
1637#  1637   1639#  1639   1641#  1641   1643#  1643   1645#  1645   1647#  1647   1649#  1649
1651#  1651   1653#  1653   1655#  1655   1657#  1657   1659#  1659   1661#  1661   1663#  1663
1665#  1665   1667#  1667   1669#  1669   1671#  1671   1673#  1673   1675#  1675   1677#  1677
1679#  1679   1681#  1681   1683#  1683   1685#  1685   1687#  1687   1689#  1689   1691#  1691
1693#  1693   1695#  1695   1697#  1697   1699#  1699   1701#  1701   1703#  1703   1705#  1705
1707#  1707   1709#  1709   1711#  1711   1713#  1713   1715#  1715   1717#  1717   1719#  1719
1721#  1721   1723#  1723   1725#  1725   1727#  1727   1729#  1729   1731#  1731   1733#  1733
1735#  1735   1737#  1737   1739#  1739   1741#  1741   1743#  1743   1745#  1745   1747#  1747
1749#  1749   1751#  1751   1753#  1753   1755#  1755   1757#  1757   1759#  1759   1761#  1761
1763#  1763   1765#  1765   1767#  1767   1769#  1769   1771#  1771   1773#  1773   1775#  1775
1777#  1777   1779#  1779   1781#  1781   1783#  1783   1785#  1785   1787#  1787   1789#  1789
1791#  1791   1793#  1793   1795#  1795   1797#  1797   1799#  1799   1801#  1801   1803#  1803
1805#  1805   1807#  1807   1809#  1809   1811#  1811   1813#  1813   1815#  1815   1817#  1817
1819#  1819   1821#  1821   1823#  1823   1825#  1825   1827#  1827   1829#  1829   1831#  1831
1833#  1833   1835#  1835   1837#  1837   1839#  1839   1841#  1841   1843#  1843   1845#  1845
1847#  1847   1849#  1849   1851#  1851   1853#  1853   1855#  1855   1857#  1857   1859#  1859
1861#  1861   1863#  1863   1865#  1865   1867#  1867   1869#  1869   1871#  1871   1873#  1873
1875#  1875   1877#  1877   1879#  1879   1881#  1881   1883#  1883   1885#  1885   1887#  1887
1889#  1889   1891#  1891   1893#  1893   1895#  1895   1897#  1897   1899#  1899   1901#  1901
1903#  1903   1905#  1905   1907#  1907   1909#  1909   1911#  1911   1913#  1913   1915#  1915
1917#  1917   1919#  1919   1921#  1921   1923#  1923   1925#  1925   1927#  1927   1929#  1929
1931#  1931   1933#  1933   1935#  1935   1937#  1937   1939#  1939   1941#  1941   1943#  1943
1945#  1945   1947#  1947   1949#  1949   1951#  1951   1953#  1953   1955#  1955   1957#  1957
1959#  1959   1961#  1961   1963#  1963   1965#  1965   1967#  1967   1969#  1969   1971#  1971
1973#  1973   1975#  1975   1977#  1977   1979#  1979   1981#  1981   1983#  1983   1985#  1985
1987#  1987   1989#  1989   1991#  1991   1993#  1993   1995#  1995   1997#  1997   1999#  1999
2001#  2001   2003#  2003   2005#  2005   2007#  2007   2009#  2009   2011#  2011   2013#  2013
2015#  2015   2017#  2017   2019#  2019   2021#  2021   2023#  2023   2025#  2025   2027#  2027
2029#  2029   2031#  2031   2033#  2033   2035#  2035   2037#  2037   2039#  2039   2041#  2041
2043#  2043   2045#  2045   2047#  2047   2049#  2049   2051#  2051   2053#  2053   2055#  2055
2057#  2057   2059#  2059   2061#  2061   2063#  2063   2065#  2065   2067#  2067   2069#  2069
```

```
.LA       2071#   2071  2073#   2073  2075#   2075  2077#   2077  2082#   2082  2087#   2087
           167    173    179    185    191    197    203    209    215    221    227    233    239    245
           251    257    263    269    275    281    287    293    299    305    311    317    323    329
           335    341    347    353    359    365    371    377    383    389    395    401    407    413
           419    425    431    437    443    449    455    461    467    473    479    485    491    497
           503    509    515    521    527    533    539    545    551    557    563    569    575    581
           587    593    599    605    611    617    623    629    635    641    647    653    659    665
           671    677    683    689    695    701    707    713    719    725    731    737    743    749
           755    761    767    773    779    785    791    797    803    809    815    821    827    833
           839    845    851    857    863    869    875    881    887    893    899    905    911    917
           923    929    935    941    947    953    959    965    971    977    983    989    995   1001
          1007   1013   1019   1025   1031   1037   1043   1049   1055   1061   1067   1073   1079   1083
          1089   1095   1101   1107   1113   1119   1125   1131   1137   1143   1149   1155   1161   1167
          1173   1179   1185   1191   1197   1203   1209   1215   1221   1227   1233   1239   1245   1251
          1257   1263   1269   1275   1281   1287   1293   1299   1305   1311   1317   1323   1329   1335
          1341   1347   1353   1359   1365   1371   1377   1383   1389   1395   1401   1407   1413   1419
          1425   1431   1437   1443   1449   1455   1461   1467   1473   1479   1485   1491   1497   1503
          1509   1515   1521   1527   1533   1539   1545   1551   1557   1563   1569   1575   1581   1587
          1593   1599   1605   1611   1617   1623   1629   1635   1641   1647   1653   1659   1665   1671
          1677   1683   1689   1695   1701   1707   1713   1719   1725   1731   1737   1743   1749   1755
          1761   1767   1773   1779   1785   1791   1797   1803   1809   1815   1821   1827   1833   1839
          1845   1851   1857   1863   1869   1875   1881   1887   1893   1899   1905   1911   1917   1923
          1929   1935   1941   1947   1953   1959   1965   1971   1977   1983   1989   1995   2001   2007
          2013   2019   2025   2031   2037   2043   2049   2055   2061   2067   2073
.LADDR     116    161   2087
.LAND      116   1119   1125   1131   1137   1143   1149   1155   1161   1167   1173   1179   1185   1191
          1197   1203   1209   1215   1221   1227   1233   1239   1245   1251   1257   1263   1269   1275
          1281   1287   1293   1299   1305   1311   1317   1323   1329   1335   1341   1347   1353
.LB        116    121    123    125    127    129    131    133    135    137    139    141    143    145
           147    149    151    161    167    173    179    185    191    197    203    209    215    221
           227    233    239    245    251    257    263    269    275    281    287    293    299    305
           311    317    323    329    335    341    347    353    359    365    371    377    383    389
           395    401    407    413    419    425    431    437    443    449    455    461    467    473
           479    485    491    497    503    509    515    521    527    533    539    545    551    557
           563    569    575    581    587    593    599    605    611    617    623    629    635    641
           647    653    659    665    671    677    683    689    695    701    707    713    719    725
           731    737    743    749    755    761    767    773    779    785    791    797    803    809
           815    821    827    833    839    845    851    857    863    869    875    881    887    893
           899    905    911    917    923    929    935    941    947    953    959    965    971    977
           983    989    995   1001   1007   1013   1019   1025   1031   1037   1043   1049   1055   1061
          1067   1073   1079   1083   1089   1095   1101   1107   1113   1119   1125   1131   1137   1143
          1149   1155   1161   1167   1173   1179   1185   1191   1197   1203   1209   1215   1221   1227
          1233   1239   1245   1251   1257   1263   1269   1275   1281   1287   1293   1299   1305   1311
          1317   1323   1329   1335   1341   1347   1353   1359   1365   1371   1377   1383   1389   1395
          1401   1407   1413   1419   1425   1431   1437   1443   1449   1455   1461   1467   1473   1479
          1485   1491   1497   1503   1509   1515   1521   1527   1533   1539   1545   1551   1557   1563
          1569   1575   1581   1587   1593   1599   1605   1611   1617   1623   1629   1635   1641   1647
          1653   1659   1665   1671   1677   1683   1689   1695   1701   1707   1713   1719   1725   1731
          1737   1743   1749   1755   1761   1767   1773   1779   1785   1791   1797   1803   1809   1815
          1821   1827   1833   1839   1845   1851   1857   1863   1869   1875   1881   1887   1893   1899
          1905   1911   1917   1923   1929   1935   1941   1947   1953   1959   1965   1971   1977   1983
          1989   1995   2001   2007   2013   2019   2025   2031   2037   2043   2049   2055   2061   2067
          2073
```

```
EQ 1144  .LBAD    2082  2087                                                                               SEQ 1157
         .LCCFZ    163   169   175   181   187   193   199   205   211   217   223   229   235   241
                   247   253   259   265   271   277   283   289   295   301   307   313   319   325
                   331   337   343   349   355   361   367   373   379   385   391   397   403   409
                   415   421   427   433   439   445   451   457   463   469   475   481   487   493
                   499   505   511   517   523   529   535   541   547   553   559   565   571   577
                   583   589   595   601   607   613   619   625   631   637   643   649   655   661
                   667   673   679   685   691   697   703   709   715   721   727   733   739   745
                   751   757   763   769   775   781   787   793   799   805   811   817   823   829
                   835   841   847   853   859   865   871   877   883   889   895   901   907   913
                   919   925   931   937   943   949   955   961   967   973   979   985   991   997
                  1003  1009  1015  1021  1027  1033  1039  1045  1051  1057  1063  1069  1075  1081
                  1085  1091  1097  1103  1109  1115  1121  1127  1133  1139  1145  1151  1157  1163
                  1169  1175  1181  1187  1193  1199  1205  1211  1217  1223  1229  1235  1241  1247
                  1253  1259  1265  1271  1277  1283  1289  1295  1301  1307  1313  1319  1325  1331
                  1337  1343  1349  1355  1361  1367  1373  1379  1385  1391  1397  1403  1409  1415
                  1421  1427  1433  1439  1445  1451  1457  1463  1469  1475  1481  1487  1493  1499
                  1505  1511  1517  1523  1529  1535  1541  1547  1553  1559  1565  1571  1577  1583
                  1589  1595  1601  1607  1613  1619  1625  1631  1637  1643  1649  1655  1661  1667
                  1673  1679  1685  1691  1697  1703  1709  1715  1721  1727  1733  1739  1745  1751
                  1757  1763  1769  1775  1781  1787  1793  1799  1805  1811  1817  1823  1829  1835
                  1841  1847  1853  1859  1865  1871  1877  1883  1889  1895  1901  1907  1913  1919
                  1925  1931  1937  1943  1949  1955  1961  1967  1973  1979  1985  1991  1997  2003
                  2009  2015  2021  2027  2033  2039  2045  2051  2057  2063  2069  2075
         .LCENA    163   169   175   181   187   193   199   205   211   217   223   229   235   241
                   247   253   259   265   271   277   283   289   295   301   307   313   319   325
                   331   337   343   349   355   361   367   373   379   385   391   397   403   409
                   415   421   427   433   439   445   451   457   463   469   475   481   487   493
                   499   505   511   517   523   529   535   541   547   553   559   565   571   577
                   583   589   595   601   607   613   619   625   631   637   643   649   655   661
                   667   673   679   685   691   697   703   709   715   721   727   733   739   745
                   751   757   763   769   775   781   787   793   799   805   811   817   823   829
                   835   841   847   853   859   865   871   877   883   889   895   901   907   913
                   919   925   931   937   943   949   955   961   967   973   979   985   991   997
                  1003  1009  1015  1021  1027  1033  1039  1045  1051  1057  1063  1069  1075  1081
                  1085  1091  1097  1103  1109  1115  1121  1127  1133  1139  1145  1151  1157  1163
                  1169  1175  1181  1187  1193  1199  1205  1211  1217  1223  1229  1235  1241  1247
                  1253  1259  1265  1271  1277  1283  1289  1295  1301  1307  1313  1319  1325  1331
                  1337  1343  1349  1355  1361  1367  1373  1379  1385  1391  1397  1403  1409  1415
                  1421  1427  1433  1439  1445  1451  1457  1463  1469  1475  1481  1487  1493  1499
                  1505  1511  1517  1523  1529  1535  1541  1547  1553  1559  1565  1571  1577  1583
                  1589  1595  1601  1607  1613  1619  1625  1631  1637  1643  1649  1655  1661  1667
                  1673  1679  1685  1691  1697  1703  1709  1715  1721  1727  1733  1739  1745  1751
                  1757  1763  1769  1775  1781  1787  1793  1799  1805  1811  1817  1823  1829  1835
                  1841  1847  1853  1859  1865  1871  1877  1883  1889  1895  1901  1907  1913  1919
                  1925  1931  1937  1943  1949  1955  1961  1967  1973  1979  1985  1991  1997  2003
                  2009  2015  2021  2027  2033  2039  2045  2051  2057  2063  2069  2075
         .LCJV     163   169   175   181   187   193   199   205   211   217   223   229   235   241
                   247   253   259   265   271   277   283   289   295   301   307   313   319   325
                   331   337   343   349   355   361   367   373   379   385   391   397   403   409
                   415   421   427   433   439   445   451   457   463   469   475   481   487   493
                   499   505   511   517   523   529   535   541   547   553   559   565   571   577
                   583   589   595   601   607   613   619   625   631   637   643   649   655   661
```

```
            667    673    679    685    691    697    703    709    715    721    727    733    739    745     SEQ 1158
            751    757    763    769    775    781    787    793    799    805    811    817    823    829
            835    841    847    853    859    865    871    877    883    889    895    901    907    913
            919    925    931    937    943    949    955    961    967    973    979    985    991    997
           1003   1009   1015   1021   1027   1033   1039   1045   1051   1057   1063   1069   1075   1081
           1085   1091   1097   1103   1109   1115   1121   1127   1133   1139   1145   1151   1157   1163
           1169   1175   1181   1187   1193   1199   1205   1211   1217   1223   1229   1235   1241   1247
           1253   1259   1265   1271   1277   1283   1289   1295   1301   1307   1313   1319   1325   1331
           1337   1343   1349   1355   1361   1367   1373   1379   1385   1391   1397   1403   1409   1415
           1421   1427   1433   1439   1445   1451   1457   1463   1469   1475   1481   1487   1493   1499
           1505   1511   1517   1523   1529   1535   1541   1547   1553   1559   1565   1571   1577   1583
           1589   1595   1601   1607   1613   1619   1625   1631   1637   1643   1649   1655   1661   1667
           1673   1679   1685   1691   1697   1703   1709   1715   1721   1727   1733   1739   1745   1751
           1757   1763   1769   1775   1781   1787   1793   1799   1805   1811   1817   1823   1829   1835
           1841   1847   1853   1859   1865   1871   1877   1883   1889   1895   1901   1907   1913   1919
           1925   1931   1937   1943   1949   1955   1961   1967   1973   1979   1985   1991   1997   2003
           2009   2015   2021   2027   2033   2039   2045   2051   2057   2063   2069   2075
.LCONT      121    123    125    127    129    131    133    135    137    139    141    143    145    147
            149    151    161    167    173    179    185    191    197    203    209    215    221    227
            233    239    245    251    257    263    269    275    281    287    293    299    305    311
            317    323    329    335    341    347    353    359    365    371    377    383    389    395
            401    407    413    419    425    431    437    443    449    455    461    467    473    479
            485    491    497    503    509    515    521    527    533    539    545    551    557    563
            569    575    581    587    593    599    605    611    617    623    629    635    641    647
            653    659    665    671    677    683    689    695    701    707    713    719    725    731
            737    743    749    755    761    767    773    779    785    791    797    803    809    815
            821    827    833    839    845    851    857    863    869    875    881    887    893    899
            905    911    917    923    929    935    941    947    953    959    965    971    977    983
            989    995   1001   1007   1013   1019   1025   1031   1037   1043   1049   1055   1061   1067
           1073   1079   1083   1089   1095   1101   1107   1113   1119   1125   1131   1137   1143   1149
           1155   1161   1167   1173   1179   1185   1191   1197   1203   1209   1215   1221   1227   1233
           1239   1245   1251   1257   1263   1269   1275   1281   1287   1293   1299   1305   1311   1317
           1323   1329   1335   1341   1347   1353   1359   1365   1371   1377   1383   1389   1395   1401
           1407   1413   1419   1425   1431   1437   1443   1449   1455   1461   1467   1473   1479   1485
           1491   1497   1503   1509   1515   1521   1527   1533   1539   1545   1551   1557   1563   1569
           1575   1581   1587   1593   1599   1605   1611   1617   1623   1629   1635   1641   1647   1653
           1659   1665   1671   1677   1683   1689   1695   1701   1707   1713   1719   1725   1731   1737
           1743   1749   1755   1761   1767   1773   1779   1785   1791   1797   1803   1809   1815   1821
           1827   1833   1839   1845   1851   1857   1863   1869   1875   1881   1887   1893   1899   1905
           1911   1917   1923   1929   1935   1941   1947   1953   1959   1965   1971   1977   1983   1989
           1995   2001   2007   2013   2019   2025   2031   2037   2043   2049   2055   2061   2067   2073
.LD         116    121    123    125    127    129    131    133    135    137    139    141    143    145
            147    149    151    153    161    163    167    169    173    175    179    181    185    187
            191    193    197    199    203    205    209    211    215    217    221    223    227    229
            233    235    239    241    245    247    251    253    257    259    263    265    269    271
            275    277    281    283    287    289    293    295    299    301    305    307    311    313
            317    319    323    325    329    331    335    337    341    343    347    349    353    355
            359    361    365    367    371    373    377    379    383    385    389    391    395    397
            401    403    407    409    413    415    419    421    425    427    431    433    437    439
            443    445    449    451    455    457    461    463    467    469    473    475    479    481
            485    487    491    493    497    499    503    505    509    511    515    517    521    523
            527    529    533    535    539    541    545    547    551    553    557    559    563    565
            569    571    575    577    581    583    587    589    593    595    599    601    605    607
```

```
            611    613    617    619    623    625    629    631    635    637    641    643    647    649
```

```
       611   613   617   619   623   625   629   631   635   637   641   643   647   649    SEQ 1159
       653   655   659   661   665   667   671   673   677   679   683   685   689   691
       695   697   701   703   707   709   713   715   719   721   725   727   731   733
       737   739   743   745   749   751   755   757   761   763   767   769   773   775
       779   781   785   787   791   793   797   799   803   805   809   811   815   817
       821   823   827   829   833   835   839   841   845   847   851   853   857   859
       863   865   869   871   875   877   881   883   887   889   893   895   899   901
       905   907   911   913   917   919   923   925   929   931   935   937   941   943
       947   949   953   955   959   961   965   967   971   973   977   979   983   985
       989   991   995   997  1001  1003  1007  1009  1013  1015  1019  1021  1025  1027
      1031  1033  1037  1039  1043  1045  1049  1051  1055  1057  1061  1063  1067  1069
      1073  1075  1079  1081  1083  1085  1089  1091  1095  1097  1101  1103  1107  1109
      1113  1115  1119  1121  1125  1127  1131  1133  1137  1139  1143  1145  1149  1151
      1155  1157  1161  1163  1167  1169  1173  1175  1179  1181  1185  1187  1191  1193
      1197  1199  1203  1205  1209  1211  1215  1217  1221  1223  1227  1229  1233  1235
      1239  1241  1245  1247  1251  1253  1257  1259  1263  1265  1269  1271  1275  1277
      1281  1283  1287  1289  1293  1295  1299  1301  1305  1307  1311  1313  1317  1319
      1323  1325  1329  1331  1335  1337  1341  1343  1347  1349  1353  1355  1359  1361
      1365  1367  1371  1373  1377  1379  1383  1385  1389  1391  1395  1397  1401  1403
      1407  1409  1413  1415  1419  1421  1425  1427  1431  1433  1437  1439  1443  1445
      1449  1451  1455  1457  1461  1463  1467  1469  1473  1475  1479  1481  1485  1487
      1491  1493  1497  1499  1503  1505  1509  1511  1515  1517  1521  1523  1527  1529
      1533  1535  1539  1541  1545  1547  1551  1553  1557  1559  1563  1565  1569  1571
      1575  1577  1581  1583  1587  1589  1593  1595  1599  1601  1605  1607  1611  1613
      1617  1619  1623  1625  1629  1631  1635  1637  1641  1643  1647  1649  1653  1655
      1659  1661  1665  1667  1671  1673  1677  1679  1683  1685  1689  1691  1695  1697
      1701  1703  1707  1709  1713  1715  1719  1721  1725  1727  1731  1733  1737  1739
      1743  1745  1749  1751  1755  1757  1761  1763  1767  1769  1773  1775  1779  1781
      1785  1787  1791  1793  1797  1799  1803  1805  1809  1811  1815  1817  1821  1823
      1827  1829  1833  1835  1839  1841  1345  1847  1851  1853  1857  1859  1863  1865
      1869  1871  1875  1877  1881  1883  1887  1889  1893  1895  1899  1901  1905  1907
      1911  1913  1917  1919  1923  1925  1929  1931  1935  1937  1941  1943  1947  1949
      1953  1955  1959  1961  1965  1967  1971  1973  1977  1979  1983  1985  1989  1991
      1995  1997  2001  2003  2007  2009  2013  2015  2019  2021  2025  2027  2031  2033
      2037  2039  2043  2045  2049  2051  2055  2057  2061  2063  2067  2069  2073  2075
.LJ    116   153   163   165   169   171   175   177   181   183   187   189   193   195
       199   201   205   207   211   213   217   219   223   225   229   231   235   237
       241   243   247   249   253   255   259   261   265   267   271   273   277   279
       283   285   289   291   295   297   301   303   307   309   313   315   319   321
       325   327   331   333   337   339   343   345   349   351   355   357   361   363
       367   369   373   375   379   381   385   387   391   393   397   399   403   405
       409   411   415   417   421   423   427   429   433   435   439   441   445   447
       451   453   457   459   463   465   469   471   475   477   481   483   487   489
       493   495   499   501   505   507   511   513   517   519   523   525   529   531
       535   537   541   543   547   549   553   555   559   561   565   567   571   573
       577   579   583   585   589   591   595   597   601   603   607   609   613   615
       619   621   625   627   631   633   637   639   643   645   649   651   655   657
       661   663   667   669   673   675   679   681   685   687   691   693   697   699
       703   705   709   711   715   717   721   723   727   729   733   735   739   741
       745   747   751   753   757   759   763   765   769   771   775   777   781   783
       787   789   793   795   799   801   805   807   811   813   817   819   823   825
       829   831   835   837   841   843   847   849   853   855   859   861   865   867
       871   873   877   879   883   885   889   891   895   897   901   903   907   909
```

```
           913   915   919   921   925   927   931   933   937   939   943   945   949   951    SEQ 1160
           955   957   961   963   967   969   973   975   979   981   985   987   991   993
           997   999  1003  1005  1009  1011  1015  1017  1021  1023  1027  1029  1033  1035
          1039  1041  1045  1047  1051  1053  1057  1059  1063  1065  1069  1071  1075  1077
          1081  1085  1087  1091  1093  1097  1099  1103  1105  1109  1111  1115  1117  1121
          1123  1127  1129  1133  1135  1139  1141  1145  1147  1151  1153  1157  1159  1163
          1165  1169  1171  1175  1177  1181  1183  1187  1189  1193  1195  1199  1201  1205
          1207  1211  1213  1217  1219  1223  1225  1229  1231  1235  1237  1241  1243  1247
          1249  1253  1255  1259  1261  1265  1267  1271  1273  1277  1279  1283  1285  1289
          1291  1295  1297  1301  1303  1307  1309  1313  1315  1319  1321  1325  1327  1331
          1333  1337  1339  1343  1345  1349  1351  1355  1357  1361  1363  1367  1369  1373
          1375  1379  1381  1385  1387  1391  1393  1397  1399  1403  1405  1409  1411  1415
          1417  1421  1423  1427  1429  1433  1435  1439  1441  1445  1447  1451  1453  1457
          1459  1463  1465  1469  1471  1475  1477  1481  1483  1487  1489  1493  1495  1499
          1501  1505  1507  1511  1513  1517  1519  1523  1525  1529  1531  1535  1537  1541
          1543  1547  1549  1553  1555  1559  1561  1565  1567  1571  1573  1577  1579  1583
          1585  1589  1591  1595  1597  1601  1603  1607  1609  1613  1615  1619  1621  1625
          1627  1631  1633  1637  1639  1643  1645  1649  1651  1655  1657  1661  1663  1667
          1669  1673  1675  1679  1681  1685  1687  1691  1693  1697  1699  1703  1705  1709
          1711  1715  1717  1721  1723  1727  1729  1733  1735  1739  1741  1745  1747  1751
          1753  1757  1759  1763  1765  1769  1771  1775  1777  1781  1783  1787  1789  1793
          1795  1799  1801  1805  1807  1811  1813  1817  1819  1823  1825  1829  1831  1835
          1837  1841  1843  1847  1849  1853  1855  1859  1861  1865  1867  1871  1873  1877
          1879  1883  1885  1889  1891  1895  1897  1901  1903  1907  1909  1913  1915  1919
          1921  1925  1927  1931  1933  1937  1939  1943  1945  1949  1951  1955  1957  1961
          1963  1967  1969  1973  1975  1979  1981  1985  1987  1991  1993  1997  1999  2003
          2005  2009  2011  2015  2017  2021  2023  2027  2029  2033  2035  2039  2041  2045
          2047  2051  2053  2057  2059  2063  2065  2069  2071  2075  2077  2082  2087
.LJMAP     116   153   165   171   177   183   189   195   201   207   213   219   225   231
           237   243   249   255   261   267   273   279   285   291   297   303   309   315
           321   327   333   339   345   351   357   363   369   375   381   387   393   399
           405   411   417   423   429   435   441   447   453   459   465   471   477   483
           489   495   501   507   513   519   525   531   537   543   549   555   561   567
           573   579   585   591   597   603   609   615   621   627   633   639   645   651
           657   663   669   675   681   687   693   699   705   711   717   723   729   735
           741   747   753   759   765   771   777   783   789   795   901   807   813   819
           825   831   837   843   849   855   861   867   873   879   885   891   897   903
           909   915   921   927   933   939   945   951   957   963   969   975   981   987
           993   999  1005  1011  1017  1023  1029  1035  1041  1047  1053  1059  1065  1071
          1077  1087  1093  1099  1105  1111  1117  1123  1129  1135  1141  1147  1153  1159
          1165  1171  1177  1183  1189  1195  1201  1207  1213  1219  1225  1231  1237  1243
          1249  1255  1261  1267  1273  1279  1285  1291  1297  1303  1309  1315  1321  1327
          1333  1339  1345  1351  1357  1363  1369  1375  1381  1387  1393  1399  1405  1411
          1417  1423  1429  1435  1441  1447  1453  1459  1465  1471  1477  1483  1489  1495
          1501  1507  1513  1519  1525  1531  1537  1543  1549  1555  1561  1567  1573  1579
          1585  1591  1597  1603  1609  1615  1621  1627  1633  1639  1645  1651  1657  1663
          1669  1675  1681  1687  1693  1699  1705  1711  1717  1723  1729  1735  1741  1747
          1753  1759  1765  1771  1777  1783  1789  1795  1801  1807  1813  1819  1825  1831
          1837  1843  1849  1855  1861  1867  1873  1879  1885  1891  1897  1903  1909  1915
          1921  1927  1933  1939  1945  1951  1957  1963  1969  1975  1981  1987  1993  1999
          2005  2011  2017  2023  2029  2035  2041  2047  2053  2059  2065  2071  2077  2082
          2087
.LMGC      121   123   125   127   129   131   133   135   137   139   141   143   145   147
```

```
              149    151
.LNAND       1359   1365   1371   1377   1383   1389   1395   1401   1407   1413   1419   1425   1431   1437
             1443   1449   1455   1461   1467   1473   1479   1485   1491   1497   1503   1509   1515   1521
             1527   1533   1539   1545   1551   1557   1563   1569   1575   1581   1587   1593
.LOR          121    123    125    127    129    131    133    135    137    139    141    143    145    147
              149    151    163    169    175    181    187    193    199    205    211    217    223    229
              235    241    247    253    259    265    271    277    283    289    295    301    307    313
              319    325    331    337    343    349    355    361    367    373    379    385    391    397
              403    409    415    421    427    433    439    445    451    457    463    469    475    481
              487    493    499    505    511    517    523    529    535    541    547    553    559    565
              571    577    583    589    595    601    607    613    619    625    631    637    643    649
              655    661    667    673    679    685    691    697    703    709    715    721    727    733
              739    745    751    757    763    769    775    781    787    793    799    805    811    817
              823    829    835    841    847    853    859    865    871    877    881    883    887    889
              893    895    899    901    905    907    911    913    917    919    923    925    929    931
              935    937    941    943    947    949    953    955    959    961    965    967    971    973
              977    979    983    985    989    991    995    997   1001   1003   1007   1009   1013   1015
             1019   1021   1025   1027   1031   1033   1037   1039   1043   1045   1049   1051   1055   1057
             1061   1063   1067   1069   1073   1075   1079   1081   1083   1085   1089   1091   1095   1097
             1101   1103   1107   1109   1113   1115   1121   1127   1133   1139   1145   1151   1157   1163
             1169   1175   1181   1187   1193   1199   1205   1211   1217   1223   1229   1235   1241   1247
             1253   1259   1265   1271   1277   1283   1289   1295   1301   1307   1313   1319   1325   1331
             1337   1343   1349   1355   1361   1367   1373   1379   1385   1391   1397   1403   1409   1415
             1421   1427   1433   1439   1445   1451   1457   1463   1469   1475   1481   1487   1493   1499
             1505   1511   1517   1523   1529   1535   1541   1547   1553   1559   1565   1571   1577   1583
             1589   1595   1601   1607   1613   1619   1625   1631   1637   1643   1649   1655   1661   1667
             1673   1679   1685   1691   1697   1703   1709   1715   1721   1727   1733   1739   1745   1751
             1757   1763   1769   1775   1781   1787   1793   1799   1805   1811   1817   1823   1829   1835
             1841   1847   1853   1859   1865   1871   1877   1883   1889   1895   1901   1907   1913   1919
             1925   1931   1937   1943   1949   1955   1961   1967   1973   1979   1985   1991   1997   2003
             2009   2015   2021   2027   2033   2039   2045   2051   2057   2063   2069   2075
.LPLUS        161    167    173    179    185    191    197    203    209    215    221    227    233    239
              245    251    257    263    269    275    281    287    293    299    305    311    317    323
              329    335    341    347    353    359    365    371    377    383    389    395
.LRMIN        641    647    653    659    665    671    677    683    689    695    701    707    713    719
              725    731    737    743    749    755    761    767    773    779    785    791    797    803
              809    815    821    827    833    839    845    851    857    863    869    875
.LSOA         281    287    293    299    305    521    527    533    539    545    761    767    773    779
              785   1001   1007   1013   1019   1025   1239   1245   1251   1257   1263   1479   1485   1491
             1497   1503   1719   1725   1731   1737   1743   1959   1965   1971   1977   1983
.LSOB         251    257    263    269    275    491    497    503    509    515    731    737    743    749
              755    971    977    983    989    995   1209   1215   1221   1227   1233   1449   1455   1461
             1467   1473   1689   1695   1701   1707   1713   1929   1935   1941   1947   1953
.LSOQ         163    169    175    181    187    193    199    205    211    217    221    223    227    229
              233    235    239    241    245    247    253    259    265    271    277    283    289    295
              301    307    313    319    325    331    337    343    349    355    361    367    373    379
              385    391    397    403    409    415    421    427    433    439    445    451    457    461
              463    467    469    473    475    479    481    485    487    493    499    505    511    517
              523    529    535    541    547    553    559    565    571    577    583    589    595    601
              607    613    619    625    631    637    643    649    655    661    667    673    679    685
              691    697    701    703    707    709    713    715    719    721    725    727    733    739
              745    751    757    763    769    775    781    787    793    799    805    811    817    823
              829    835    841    847    853    859    865    871    877    883    889    895    901    907
```

```
           913   919   925   931   937   941   943   947   949   953   955   959   961   965    SEQ 1162
           967   973   979   985   991   997  1003  1009  1015  1021  1027  1033  1039  1045
          1051  1057  1063  1069  1075  1081  1085  1091  1097  1103  1109  1115  1121  1127
          1133  1139  1145  1151  1157  1163  1169  1175  1179  1181  1185  1187  1191  1193
          1197  1199  1203  1205  1211  1217  1223  1229  1235  1241  1247  1253  1259  1265
          1271  1277  1283  1289  1295  1301  1307  1313  1319  1325  1331  1337  1343  1349
          1355  1361  1367  1373  1379  1385  1391  1397  1403  1409  1415  1419  1421  1425
          1427  1431  1433  1437  1439  1443  1445  1451  1457  1463  1469  1475  1481  1487
          1493  1499  1505  1511  1517  1523  1529  1535  1541  1547  1553  1559  1565  1571
          1577  1583  1589  1595  1601  1607  1613  1619  1625  1631  1637  1643  1649  1655
          1659  1661  1665  1667  1671  1673  1677  1679  1683  1685  1691  1697  1703  1709
          1715  1721  1727  1733  1739  1745  1751  1757  1763  1769  1775  1781  1787  1793
          1799  1805  1811  1817  1823  1829  1835  1841  1847  1853  1859  1865  1871  1877
          1883  1889  1895  1899  1901  1905  1907  1911  1913  1917  1919  1923  1925  1931
          1937  1943  1949  1955  1961  1967  1973  1979  1985  1991  1997  2003  2009  2015
          2021  2027  2033  2039  2045  2051  2057  2063  2069  2075
.LSAB      191   197   203   209   215   431   437   443   449   455   671   677   683   689
           695   911   917   923   929   935  1149  1155  1161  1167  1173  1389  1395  1401
          1407  1413  1629  1635  1641  1647  1653  1869  1875  1881  1887  1893
.LSAQ      161   167   173   179   185   401   407   413   419   425   641   647   653   659
           665   881   887   893   899   905  1119  1125  1131  1137  1143  1359  1365  1371
          1377  1383  1599  1605  1611  1617  1623  1839  1845  1851  1857  1863
.LSDO      116   121   123   125   127   129   131   133   135   137   139   141   143   145
           147   149   151   371   377   383   389   395   611   617   623   629   635   851
           857   863   869   875  1089  1095  1101  1107  1113  1329  1335  1341  1347  1353
          1569  1575  1581  1587  1593  1809  1815  1821  1827  1833  2049  2055  2061  2067
          2073
.LSDA      311   317   323   329   335   551   557   563   569   575   791   797   803   809
           815  1031  1037  1043  1049  1055  1269  1275  1281  1287  1293  1509  1515  1521
          1527  1533  1749  1755  1761  1767  1773  1989  1995  2001  2007  2013
.LSDQ      341   347   353   359   365   581   587   593   599   605   821   827   833   839
           845  1061  1067  1073  1079  1083  1299  1305  1311  1317  1323  1539  1545  1551
          1557  1563  1779  1785  1791  1797  1803  2019  2025  2031  2037  2043
.LSKCN     121   123   125   127   129   131   133   135   137   139   141   143   145   147
           149   151
.LSMIN     401   407   413   419   425   431   437   443   449   455   461   467   473   479
           485   491   497   503   509   515   521   527   533   539   545   551   557   563
           569   575   581   587   593   599   605   611   617   623   629   635
.LXNOR    1839  1845  1851  1857  1863  1869  1875  1881  1887  1893  1899  1905  1911  1917
          1923  1929  1935  1941  1947  1953  1959  1965  1971  1977  1983  1989  1995  2001
          2007  2013  2019  2025  2031  2037  2043  2049  2055  2061  2067  2073
.LXOR     1599  1605  1611  1617  1623  1629  1635  1641  1647  1653  1659  1665  1671  1677
          1683  1689  1695  1701  1707  1713  1719  1725  1731  1737  1743  1749  1755  1761
          1767  1773  1779  1785  1791  1797  1803  1809  1815  1821  1827  1833
.MA        167   173   179   185   191   197   203   209   215   221   227   233   239   245
           251   257   263   269   275   281   287   293   299   305   311   317   323   329
           335   341   347   353   359   365   371   377   383   389   395   401   407   413
           419   425   431   437   443   449   455   461   467   473   479   485   491   497
           503   509   515   521   527   533   539   545   551   557   563   569   575   581
           587   593   599   605   611   617   623   629   635   641   647   653   659   665
           671   677   683   689   695   701   707   713   719   725   731   737   743   749
           755   761   767   773   779   785   791   797   803   809   815   821   827   833
           839   845   851   857   863   869   875   881   887   893   899   905   911   917
           923   929   935   941   947   953   959   965   971   977   983   989   995  1001    SEQ 1163
          1007  1013  1019  1025  1031  1037  1043  1049  1055  1061  1067  1073  1079  1083
          1089  1095  1101  1107  1113  1119  1125  1131  1137  1143  1149  1155  1161  1167
          1173  1179  1185  1191  1197  1203  1209  1215  1221  1227  1233  1239  1245  1251
          1257  1263  1269  1275  1281  1287  1293  1299  1305  1311  1317  1323  1329  1335
          1341  1347  1353  1359  1365  1371  1377  1383  1389  1395  1401  1407  1413  1419
          1425  1431  1437  1443  1449  1455  1461  1467  1473  1479  1485  1491  1497  1503
          1509  1515  1521  1527  1533  1539  1545  1551  1557  1563  1569  1575  1581  1587
          1593  1599  1605  1611  1617  1623  1629  1635  1641  1647  1653  1659  1665  1671
          1677  1683  1689  1695  1701  1707  1713  1719  1725  1731  1737  1743  1749  1755
          1761  1767  1773  1779  1785  1791  1797  1803  1809  1815  1821  1827  1833  1839
          1845  1851  1857  1863  1869  1875  1881  1887  1893  1899  1905  1911  1917  1923
          1929  1935  1941  1947  1953  1959  1965  1971  1977  1983  1989  1995  2001  2007
          2013  2019  2025  2031  2037  2043  2049  2055  2061  2067  2073
.MAND      116  1119  1125  1131  1137  1143  1149  1155  1161  1167  1173  1179  1185  1191
          1197  1203  1209  1215  1221  1227  1233  1239  1245  1251  1257  1263  1269  1275
          1281  1287  1293  1299  1305  1311  1317  1323  1329  1335  1341  1347  1353
.MB        116   121   123   125   127   129   131   133   135   137   139   141   143   145
           147   149   151   161   167   173   179   185   191   197   203   209   215   221
           227   233   239   245   251   257   263   269   275   281   287   293   299   305
           311   317   323   329   335   341   347   353   359   365   371   377   383   389
           395   401   407   413   419   425   431   437   443   449   455   461   467   473
           479   485   491   497   503   509   515   521   527   533   539   545   551   557
           563   569   575   581   587   593   599   605   611   617   623   629   635   641
           647   653   659   665   671   677   683   689   695   701   707   713   719   725
           731   737   743   749   755   761   767   773   779   785   791   797   803   809
           815   821   827   833   839   845   851   857   863   869   875   881   887   893
           899   905   911   917   923   929   935   941   947   953   959   965   971   977
           983   989   995  1001  1007  1013  1019  1025  1031  1037  1043  1049  1055  1061
          1067  1073  1079  1083  1089  1095  1101  1107  1113  1119  1125  1131  1137  1143
          1149  1155  1161  1167  1173  1179  1185  1191  1197  1203  1209  1215  1221  1227
          1233  1239  1245  1251  1257  1263  1269  1275  1281  1287  1293  1299  1305  1311
          1317  1323  1329  1335  1341  1347  1353  1359  1365  1371  1377  1383  1389  1395
          1401  1407  1413  1419  1425  1431  1437  1443  1449  1455  1461  1467  1473  1479
          1485  1491  1497  1503  1509  1515  1521  1527  1533  1539  1545  1551  1557  1563
          1569  1575  1581  1587  1593  1599  1605  1611  1617  1623  1629  1635  1641  1647
          1653  1659  1665  1671  1677  1683  1689  1695  1701  1707  1713  1719  1725  1731
          1737  1743  1749  1755  1761  1767  1773  1779  1785  1791  1797  1803  1809  1815
          1821  1827  1833  1839  1845  1851  1857  1863  1869  1875  1881  1887  1893  1899
          1905  1911  1917  1923  1929  1935  1941  1947  1953  1959  1965  1971  1977  1983
          1989  1995  2001  2007  2013  2019  2025  2031  2037  2043  2049  2055  2061  2067
          2073
.MBAD     2082  2087
.MCCFZ     163   169   175   181   187   193   199   205   211   217   223   229   235   241
           247   253   259   265   271   277   283   289   295   301   307   313   319   325
           331   337   343   349   355   361   367   373   379   385   391   397   403   409
           415   421   427   433   439   445   451   457   463   469   475   481   487   493
           499   505   511   517   523   529   535   541   547   553   559   565   571   577
           583   589   595   601   607   613   619   625   631   637   643   649   655   661
           667   673   679   685   691   697   703   709   715   721   727   733   739   745
           751   757   763   769   775   781   787   793   799   805   811   817   823   829
           835   841   847   853   859   865   871   877   883   889   895   901   907   913
           919   925   931   937   943   949   955   961   967   973   979   985   991   997
          1003  1009  1015  1021  1027  1033  1039  1045  1051  1057  1063  1069  1075  1081
          1085  1091  1097  1103  1109  1115  1121  1127  1133  1139  1145  1151  1157  1163
          1169  1175  1181  1187  1193  1199  1205  1211  1217  1223  1229  1235  1241  1247
          1253  1259  1265  1271  1277  1283  1289  1295  1301  1307  1313  1319  1325  1331
          1337  1343  1349  1355  1361  1367  1373  1379  1385  1391  1397  1403  1409  1415
          1421  1427  1433  1439  1445  1451  1457  1463  1469  1475  1481  1487  1493  1499
          1505  1511  1517  1523  1529  1535  1541  1547  1553  1559  1565  1571  1577  1583
          1589  1595  1601  1607  1613  1619  1625  1631  1637  1643  1649  1655  1661  1667
          1673  1679  1685  1691  1697  1703  1709  1715  1721  1727  1733  1739  1745  1751
          1757  1763  1769  1775  1781  1787  1793  1799  1805  1811  1817  1823  1829  1835
          1841  1847  1853  1859  1865  1871  1877  1883  1889  1895  1901  1907  1913  1919
          1925  1931  1937  1943  1949  1955  1961  1967  1973  1979  1985  1991  1997  2003
          2009  2015  2021  2027  2033  2039  2045  2051  2057  2063  2069  2075
.MCENA     163   169   175   181   187   193   199   205   211   217   223   229   235   241
           247   253   259   265   271   277   283   289   295   301   307   313   319   325
           331   337   343   349   355   361   367   373   379   385   391   397   403   409
           415   421   427   433   439   445   451   457   463   469   475   481   487   493
           499   505   511   517   523   529   535   541   547   553   559   565   571   577
           583   589   595   601   607   613   619   625   631   637   643   649   655   661
           667   673   679   685   691   697   703   709   715   721   727   733   739   745
           751   757   763   769   775   781   787   793   799   805   811   817   823   829
           835   841   847   853   859   865   871   877   883   889   895   901   907   913
           919   925   931   937   943   949   955   961   967   973   979   985   991   997
          1003  1009  1015  1021  1027  1033  1039  1045  1051  1057  1063  1069  1075  1081
          1085  1091  1097  1103  1109  1115  1121  1127  1133  1139  1145  1151  1157  1163
          1169  1175  1181  1187  1193  1199  1205  1211  1217  1223  1229  1235  1241  1247
          1253  1259  1265  1271  1277  1283  1289  1295  1301  1307  1313  1319  1325  1331
          1337  1343  1349  1355  1361  1367  1373  1379  1385  1391  1397  1403  1409  1415
          1421  1427  1433  1439  1445  1451  1457  1463  1469  1475  1481  1487  1493  1499
          1505  1511  1517  1523  1529  1535  1541  1547  1553  1559  1565  1571  1577  1583
          1589  1595  1601  1607  1613  1619  1625  1631  1637  1643  1649  1655  1661  1667
          1673  1679  1685  1691  1697  1703  1709  1715  1721  1727  1733  1739  1745  1751
          1757  1763  1769  1775  1781  1787  1793  1799  1805  1811  1817  1823  1829  1835
          1841  1847  1853  1859  1865  1871  1877  1883  1889  1895  1901  1907  1913  1919
          1925  1931  1937  1943  1949  1955  1961  1967  1973  1979  1985  1991  1997  2003
          2009  2015  2021  2027  2033  2039  2045  2051  2057  2063  2069  2075
.MCJV      163   169   175   181   187   193   199   205   211   217   223   229   235   241
           247   253   259   265   271   277   283   289   295   301   307   313   319   325
           331   337   343   349   355   361   367   373   379   385   391   397   403   409
           415   421   427   433   439   445   451   457   463   469   475   481   487   493
           499   505   511   517   523   529   535   541   547   553   559   565   571   577
           583   589   595   601   607   613   619   625   631   637   643   649   655   661
           667   673   679   685   691   697   703   709   715   721   727   733   739   745
           751   757   763   769   775   781   787   793   799   805   811   817   823   829
           835   841   847   853   859   865   871   877   883   889   895   901   907   913
           919   925   931   937   943   949   955   961   967   973   979   985   991   997
          1003  1009  1015  1021  1027  1033  1039  1045  1051  1057  1063  1069  1075  1081
          1085  1091  1097  1103  1109  1115  1121  1127  1133  1139  1145  1151  1157  1163
          1169  1175  1181  1187  1193  1199  1205  1211  1217  1223  1229  1235  1241  1247
          1253  1259  1265  1271  1277  1283  1289  1295  1301  1307  1313  1319  1325  1331
          1337  1343  1349  1355  1361  1367  1373  1379  1385  1391  1397  1403  1409  1415
          1421  1427  1433  1439  1445  1451  1457  1463  1469  1475  1481  1487  1493  1499
          1505  1511  1517  1523  1529  1535  1541  1547  1553  1559  1565  1571  1577  1583
```

```
        1589  1595  1601  1607  1613  1619  1625  1631  1637  1643  1649  1655  1661  1667
        1673  1679  1685  1691  1697  1703  1709  1715  1721  1727  1733  1739  1745  1751
        1757  1763  1769  1775  1781  1787  1793  1799  1805  1811  1817  1823  1829  1835
        1841  1847  1853  1859  1865  1871  1877  1883  1889  1895  1901  1907  1913  1919
        1925  1931  1937  1943  1949  1955  1961  1967  1973  1979  1985  1991  1997  2003
        2009  2015  2021  2027  2033  2039  2045  2051  2057  2063  2069  2075
.MCONT   121   123   125   127   129   131   133   135   137   139   141   143   145   147
         149   151   161   167   173   179   185   191   197   203   209   215   221   227
         233   239   245   251   257   263   269   275   281   287   293   299   305   311
         317   323   329   335   341   347   353   359   365   371   377   383   389   395
         401   407   413   419   425   431   437   443   449   455   461   467   473   479
         485   491   497   503   509   515   521   527   533   539   545   551   557   563
         569   575   581   587   593   599   605   611   617   623   629   635   641   647
         653   659   665   671   677   683   689   695   701   707   713   719   725   731
         737   743   749   755   761   767   773   779   785   791   797   803   809   815
         821   827   833   839   845   851   857   863   869   875   881   887   893   899
         905   911   917   923   929   935   941   947   953   959   965   971   977   983
         989   995  1001  1007  1013  1019  1025  1031  1037  1043  1049  1055  1061  1067
        1073  1079  1083  1089  1095  1101  1107  1113  1119  1125  1131  1137  1143  1149
        1155  1161  1167  1173  1179  1185  1191  1197  1203  1209  1215  1221  1227  1233
        1239  1245  1251  1257  1263  1269  1275  1281  1287  1293  1299  1305  1311  1317
        1323  1329  1335  1341  1347  1353  1359  1365  1371  1377  1383  1389  1395  1401
        1407  1413  1419  1425  1431  1437  1443  1449  1455  1461  1467  1473  1479  1485
        1491  1497  1503  1509  1515  1521  1527  1533  1539  1545  1551  1557  1563  1569
        1575  1581  1587  1593  1599  1605  1611  1617  1623  1629  1635  1641  1647  1653
        1659  1665  1671  1677  1683  1689  1695  1701  1707  1713  1719  1725  1731  1737
        1743  1749  1755  1761  1767  1773  1779  1785  1791  1797  1803  1809  1815  1821
        1827  1833  1839  1845  1851  1857  1863  1869  1875  1881  1887  1893  1899  1905
        1911  1917  1923  1929  1935  1941  1947  1953  1959  1965  1971  1977  1983  1989
        1995  2001  2007  2013  2019  2025  2031  2037  2043  2049  2055  2061  2067  2073
.MD      116   121   123   125   127   129   131   133   135   137   139   141   143   145
         147   149   151   153   161   163   167   169   173   175   179   181   185   187
         191   193   197   199   203   205   209   211   215   217   221   223   227   229
         233   235   239   241   245   247   251   253   257   259   263   265   269   271
         275   277   281   283   287   289   293   295   299   301   305   307   311   313
         317   319   323   325   329   331   335   337   341   343   347   349   353   355
         359   361   365   367   371   373   377   379   383   385   389   391   395   397
         401   403   407   409   413   415   419   421   425   427   431   433   437   439
         443   445   449   451   455   457   461   463   467   469   473   475   479   481
         485   487   491   493   497   499   503   505   509   511   515   517   521   523
         527   529   533   535   539   541   545   547   551   553   557   559   563   565
         569   571   575   577   581   583   587   589   593   595   599   601   605   607
         611   613   617   619   623   625   629   631   635   637   641   643   647   649
         653   655   659   661   665   667   671   673   677   679   683   685   689   691
         695   697   701   703   707   709   713   715   719   721   725   727   731   733
         737   739   743   745   749   751   755   757   761   763   767   769   773   775
         779   781   785   787   791   793   797   799   803   805   809   811   815   617
         821   823   827   829   833   835   839   841   845   847   851   853   857   859
         863   865   869   871   875   877   881   883   887   889   893   895   899   901
         905   907   911   913   917   919   923   925   929   931   935   937   941   943
         947   949   953   955   959   961   965   967   971   973   977   979   983   985
         989   991   995   997  1001  1003  1007  1009  1013  1015  1019  1021  1025  1027
        1031  1033  1037  1039  1043  1045  1049  1051  1055  1057  1061  1063  1067  1069
```

1153

SEQ 1166

.MJ

1073	1075	1079	1081	1083	1085	1089	1091	1095	1097	1101	1103	1107	1109
1113	1115	1119	1121	1125	1127	1131	1133	1137	1139	1143	1145	1149	1151
1155	1157	1161	1163	1167	1169	1173	1175	1179	1181	1185	1187	1191	1193
1197	1199	1203	1205	1209	1211	1215	1217	1221	1223	1227	1229	1233	1235
1239	1241	1245	1247	1251	1253	1257	1259	1263	1265	1269	1271	1275	1277
1281	1283	1287	1289	1293	1295	1299	1301	1305	1307	1311	1313	1317	1319
1323	1325	1329	1331	1335	1337	1341	1343	1347	1349	1353	1355	1359	1361
1365	1367	1371	1373	1377	1379	1383	1385	1389	1391	1395	1397	1401	1403
1407	1409	1413	1415	1419	1421	1425	1427	1431	1433	1437	1439	1443	1445
1449	1451	1455	1457	1461	1463	1467	1469	1473	1475	1479	1481	1485	1487
1491	1493	1497	1499	1503	1505	1509	1511	1515	1517	1521	1523	1527	1529
1533	1535	1539	1541	1545	1547	1551	1553	1557	1559	1563	1565	1569	1571
1575	1577	1581	1583	1587	1589	1593	1595	1599	1601	1605	1607	1611	1613
1617	1619	1623	1625	1629	1631	1635	1637	1641	1643	1647	1649	1653	1655
1659	1661	1665	1667	1671	1673	1677	1679	1683	1685	1689	1691	1695	1697
1701	1703	1707	1709	1713	1715	1719	1721	1725	1727	1731	1733	1737	1739
1743	1745	1749	1751	1755	1757	1761	1763	1767	1769	1773	1775	1779	1781
1785	1787	1791	1793	1797	1799	1803	1805	1809	1811	1815	1817	1821	1823
1827	1829	1833	1835	1839	1841	1845	1847	1851	1853	1857	1859	1863	1865
1869	1871	1875	1877	1881	1883	1887	1889	1893	1895	1899	1901	1905	1907
1911	1913	1917	1919	1923	1925	1929	1931	1935	1937	1941	1943	1947	1949
1953	1955	1959	1961	1965	1967	1971	1973	1977	1979	1983	1985	1989	1991
1995	1997	2001	2003	2007	2009	2013	2015	2019	2021	2025	2027	2031	2033
2037	2039	2043	2045	2049	2051	2055	2057	2061	2063	2067	2069	2073	2075
116	153	163	165	169	171	175	177	181	183	187	189	193	195
199	201	205	207	211	213	217	219	223	225	229	231	235	237
241	243	247	249	253	255	259	261	265	267	271	273	277	279
283	285	289	291	295	297	301	303	307	309	313	315	319	321
325	327	331	333	337	339	343	345	349	351	355	357	361	363
367	369	373	375	379	381	385	387	391	393	397	399	403	405
409	411	415	417	421	423	427	429	433	435	439	441	445	447
451	453	457	459	463	465	469	471	475	477	481	483	487	489
493	495	499	501	505	507	511	513	517	519	523	525	529	531
535	537	541	543	547	549	553	555	559	561	565	567	571	573
577	579	583	585	589	591	595	597	601	603	607	609	613	615
619	621	625	627	631	633	637	639	643	645	649	651	655	657
661	663	667	669	673	675	679	681	685	687	691	693	697	699
703	705	709	711	715	717	721	723	727	729	733	735	739	741
745	747	751	753	757	759	763	765	769	771	775	777	781	783
787	789	793	795	799	801	805	807	811	813	817	819	823	825
829	831	835	837	841	843	847	849	853	855	859	861	865	867
871	873	877	879	883	885	889	891	895	897	901	903	907	909
913	915	919	921	925	927	931	933	937	939	943	945	949	951
955	957	961	963	967	969	973	975	979	981	985	987	991	993
997	999	1003	1005	1009	1011	1015	1017	1021	1023	1027	1029	1033	1035
1039	1041	1045	1047	1051	1053	1057	1059	1063	1065	1069	1071	1075	1077
1081	1085	1087	1091	1093	1097	1099	1103	1105	1109	1111	1115	1117	1121
1123	1127	1129	1133	1135	1139	1141	1145	1147	1151	1153	1157	1159	1163
1165	1169	1171	1175	1177	1181	1183	1187	1189	1193	1195	1199	1201	1205
1207	1211	1213	1217	1219	1223	1225	1229	1231	1235	1237	1241	1243	1247
1249	1253	1255	1259	1261	1265	1267	1271	1273	1277	1279	1283	1285	1289
1291	1295	1297	1301	1303	1307	1309	1313	1315	1319	1321	1325	1327	1331
1333	1337	1339	1343	1345	1349	1351	1355	1357	1361	1363	1367	1369	1373

```
             1375   1379   1381   1385   1387   1391   1393   1397   1399   1403   1405   1409   1411   1415
             1417   1421   1423   1427   1429   1433   1435   1439   1441   1445   1447   1451   1453   1457
             1459   1463   1465   1469   1471   1475   1477   1481   1483   1487   1489   1493   1495   1499
             1501   1505   1507   1511   1513   1517   1519   1523   1525   1529   1531   1535   1537   1541
             1543   1547   1549   1553   1555   1559   1561   1565   1567   1571   1573   1577   1579   1583
             1585   1589   1591   1595   1597   1601   1603   1607   1609   1613   1615   1619   1621   1625
             1627   1631   1633   1637   1639   1643   1645   1649   1651   1655   1657   1661   1663   1667
             1669   1673   1675   1679   1681   1685   1687   1691   1693   1697   1699   1703   1705   1709
             1711   1715   1717   1721   1723   1727   1729   1733   1735   1739   1741   1745   1747   1751
             1753   1757   1759   1763   1765   1769   1771   1775   1777   1781   1783   1787   1789   1793
             1795   1799   1801   1805   1807   1811   1813   1817   1819   1823   1825   1829   1831   1835
             1837   1841   1843   1847   1849   1853   1855   1859   1861   1865   1867   1871   1873   1877
             1879   1883   1885   1889   1891   1895   1897   1901   1903   1907   1909   1913   1915   1919
             1921   1925   1927   1931   1933   1937   1939   1943   1945   1949   1951   1955   1957   1961
             1963   1967   1969   1973   1975   1979   1981   1985   1987   1991   1993   1997   1999   2003
             2005   2009   2011   2015   2017   2021   2023   2027   2029   2033   2035   2039   2041   2045
             2047   2051   2053   2057   2059   2063   2065   2069   2071   2075   2077   2082   2087
.MJMAP        116    153    165    171    177    183    189    195    201    207    213    219    225    231
              237    243    249    255    261    267    273    279    285    291    297    303    309    315
              321    327    333    339    345    351    357    363    369    375    381    387    393    399
              405    411    417    423    429    435    441    447    453    459    465    471    477    483
              489    495    501    507    513    519    525    531    537    543    549    555    561    567
              573    579    585    591    597    603    609    615    621    627    633    639    645    651
              657    663    669    675    681    687    693    699    705    711    717    723    729    735
              741    747    753    759    765    771    777    783    789    795    801    807    813    819
              825    831    837    843    849    855    861    867    873    879    885    891    897    903
              909    915    921    927    933    939    945    951    957    963    969    975    981    987
              993    999   1005   1011   1017   1023   1029   1035   1041   1047   1053   1059   1065   1071
             1077   1087   1093   1099   1105   1111   1117   1123   1129   1135   1141   1147   1153   1159
             1165   1171   1177   1183   1189   1195   1201   1207   1213   1219   1225   1231   1237   1243
             1249   1255   1261   1267   1273   1279   1285   1291   1297   1303   1309   1315   1321   1327
             1333   1339   1345   1351   1357   1363   1369   1375   1381   1387   1393   1399   1405   1411
             1417   1423   1429   1435   1441   1447   1453   1459   1465   1471   1477   1483   1489   1495
             1501   1507   1513   1519   1525   1531   1537   1543   1549   1555   1561   1567   1573   1579
             1585   1591   1597   1603   1609   1615   1621   1627   1633   1639   1645   1651   1657   1663
             1669   1675   1681   1687   1693   1699   1705   1711   1717   1723   1729   1735   1741   1747
             1753   1759   1765   1771   1777   1783   1789   1795   1801   1807   1813   1819   1825   1831
             1837   1843   1849   1855   1861   1867   1873   1879   1885   1891   1897   1903   1909   1915
             1921   1927   1933   1939   1945   1951   1957   1963   1969   1975   1981   1987   1993   1999
             2005   2011   2017   2023   2029   2035   2041   2047   2053   2059   2065   2071   2077   2082
             2087
.MMGC         121    123    125    127    129    131    133    135    137    139    141    143    145    147
              149    151
.MNAND       1359   1365   1371   1377   1383   1389   1395   1401   1407   1413   1419   1425   1431   1437
             1443   1449   1455   1461   1467   1473   1479   1485   1491   1497   1503   1509   1515   1521
             1527   1533   1539   1545   1551   1557   1563   1569   1575   1581   1587   1593
.MOR          121    123    125    127    129    131    133    135    137    139    141    143    145    147
              149    151    163    169    175    181    187    193    199    205    211    217    223    229
              235    241    247    253    259    265    271    277    283    289    295    301    307    313
              319    325    331    337    343    349    355    361    367    373    379    385    391    397
              403    409    415    421    427    433    439    445    451    457    463    469    475    481
              487    493    499    505    511    517    523    529    535    541    547    553    559    565
              571    577    583    589    595    601    607    613    619    625    631    637    643    649
```

```
           655   661   667   673   679   685   691   697   703   709   715   721   727   733
           739   745   751   757   763   769   775   781   787   793   799   805   811   817
           823   829   835   841   847   853   859   865   871   877   881   883   887   889
           893   895   899   901   905   907   911   913   917   919   923   925   929   931
           935   937   941   943   947   949   953   955   959   961   965   967   971   973
           977   979   983   985   989   991   995   997  1001  1003  1007  1009  1013  1015
          1019  1021  1025  1027  1031  1033  1037  1039  1043  1045  1049  1051  1055  1057
          1061  1063  1067  1069  1073  1075  1079  1081  1083  1085  1089  1091  1095  1097
          1101  1103  1107  1109  1113  1115  1121  1127  1133  1139  1145  1151  1157  1163
          1169  1175  1181  1187  1193  1199  1205  1211  1217  1223  1229  1235  1241  1247
          1253  1259  1265  1271  1277  1283  1289  1295  1301  1307  1313  1319  1325  1331
          1337  1343  1349  1355  1361  1367  1373  1379  1385  1391  1397  1403  1409  1415
          1421  1427  1433  1439  1445  1451  1457  1463  1469  1475  1481  1487  1493  1499
          1505  1511  1517  1523  1529  1535  1541  1547  1553  1559  1565  1571  1577  1583
          1589  1595  1601  1607  1613  1619  1625  1631  1637  1643  1649  1655  1661  1667
          1673  1679  1685  1691  1697  1703  1709  1715  1721  1727  1733  1739  1745  1751
          1757  1763  1769  1775  1781  1787  1793  1799  1805  1811  1817  1823  1829  1835
          1841  1847  1853  1859  1865  1871  1877  1883  1889  1895  1901  1907  1913  1919
          1925  1931  1937  1943  1949  1955  1961  1967  1973  1979  1985  1991  1997  2003
          2009  2015  2021  2027  2033  2039  2045  2051  2057  2063  2069  2075
.MPLUS     161   167   173   179   185   191   197   203   209   215   221   227   233   239
           245   251   257   263   269   275   281   287   293   299   305   311   317   323
           329   335   341   347   353   359   365   371   377   383   389   395
.MRMIN     641   647   653   659   665   671   677   683   689   695   701   707   713   719
           725   731   737   743   749   755   761   767   773   779   785   791   797   803
           809   815   821   827   833   839   845   851   857   863   869   875
.MSOA      281   287   293   299   305   521   527   533   539   545   761   767   773   779
           785  1001  1007  1013  1019  1025  1239  1245  1251  1257  1263  1479  1485  1491
          1497  1503  1719  1725  1731  1737  1743  1959  1965  1971  1977  1983
.MSOB      251   257   263   269   275   491   497   503   509   515   731   737   743   749
           755   971   977   983   989   995  1209  1215  1221  1227  1233  1449  1455  1461
          1467  1473  1689  1695  1701  1707  1713  1929  1935  1941  1947  1953
.MSOQ      163   169   175   181   187   193   199   205   211   217   221   223   227   229
           233   235   239   241   245   247   253   259   265   271   277   283   289   295
           301   307   313   319   325   331   337   343   349   355   361   367   373   379
           385   391   397   403   409   415   421   427   433   439   445   451   457   461
           463   467   469   473   475   479   481   485   487   493   499   505   511   517
           523   529   535   541   547   553   559   565   571   577   583   589   595   601
           607   613   619   625   631   637   643   649   655   661   667   673   679   685
           691   697   701   703   707   709   713   715   719   721   725   727   733   739
           745   751   757   763   769   775   781   787   793   799   805   811   817   823
           829   835   841   847   853   859   865   871   877   883   889   895   901   907
           913   919   925   931   937   941   943   947   949   953   955   959   961   965
           967   973   979   985   991   997  1003  1009  1015  1021  1027  1033  1039  1045
          1051  1057  1063  1069  1075  1081  1085  1091  1097  1103  1109  1115  1121  1127
          1133  1139  1145  1151  1157  1163  1169  1175  1179  1181  1185  1187  1191  1193
          1197  1199  1203  1205  1211  1217  1223  1229  1235  1241  1247  1253  1259  1265
          1271  1277  1283  1289  1295  1301  1307  1313  1319  1325  1331  1337  1343  1349
          1355  1361  1367  1373  1379  1385  1391  1397  1403  1409  1415  1419  1421  1425
          1427  1431  1433  1437  1439  1443  1445  1451  1457  1463  1469  1475  1481  1487
          1493  1499  1505  1511  1517  1523  1529  1535  1541  1547  1553  1559  1565  1571
          1577  1583  1589  1595  1601  1607  1613  1619  1625  1631  1637  1643  1649  1655
          1659  1661  1665  1667  1671  1673  1677  1679  1683  1685  1691  1697  1703  1709
          1715  1721  1727  1733  1739  1745  1751  1757  1763  1769  1775  1781  1787  1793
          1799  1805  1811  1817  1823  1829  1835  1841  1847  1853  1859  1865  1871  1877
          1883  1889  1895  1899  1901  1905  1907  1911  1913  1917  1919  1923  1925  1931
          1937  1943  1949  1955  1961  1967  1973  1979  1985  1991  1997  2003  2009  2015
          2021  2027  2033  2039  2045  2051  2057  2063  2069  2075
.MSAB      191   197   203   209   215   431   437   443   449   455   671   677   683   689
           695   911   917   923   929   935  1149  1155  1161  1167  1173  1389  1395  1401
          1407  1413  1629  1635  1641  1647  1653  1869  1875  1881  1887  1893
.MSAQ      161   167   173   179   185   401   407   413   419   425   641   647   653   659
           665   881   887   893   899   905  1119  1125  1131  1137  1143  1359  1365  1371
          1377  1383  1599  1605  1611  1617  1623  1839  1845  1851  1857  1863
.MSDO      116   121   123   125   127   129   131   133   135   137   139   141   143   145
           147   149   151   371   377   383   389   395   611   617   623   629   635   851
           857   863   869   875  1089  1095  1101  1107  1113  1329  1335  1341  1347  1353
          1569  1575  1581  1587  1593  1809  1815  1821  1827  1833  2049  2055  2061  2067
          2073
.MSDA      311   317   323   329   335   551   557   563   569   575   791   797   803   809
           815  1031  1037  1043  1049  1055  1269  1275  1281  1287  1293  1509  1515  1521
          1527  1533  1749  1755  1761  1767  1773  1989  1995  2001  2007  2013
.MSDQ      341   347   353   359   365   581   587   593   599   605   821   827   833   839
           845  1061  1067  1073  1079  1083  1299  1305  1311  1317  1323  1539  1545  1551
          1557  1563  1779  1785  1791  1797  1803  2019  2025  2031  2037  2043
.MSKCN     121   123   125   127   129   131   133   135   137   139   141   143   145   147
           149   151
.MSMIN     401   407   413   419   425   431   437   443   449   455   461   467   473   479
           485   491   497   503   509   515   521   527   533   539   545   551   557   563
           569   575   581   587   593   599   605   611   617   623   629   635
.MXNOR    1839  1845  1851  1857  1863  1869  1875  1881  1887  1893  1899  1905  1911  1917
          1923  1929  1935  1941  1947  1953  1959  1965  1971  1977  1983  1989  1995  2001
          2007  2013  2019  2025  2031  2037  2043  2049  2055  2061  2067  2073
.MXOR     1599  1605  1611  1617  1623  1629  1635  1641  1647  1653  1659  1665  1671  1677
          1683  1689  1695  1701  1707  1713  1719  1725  1731  1737  1743  1749  1755  1761
          1767  1773  1779  1785  1791  1797  1803  1809  1815  1821  1827  1833
.RA        167   173   179   185   191   197   203   209   215   221   227   233   239   245
           251   257   263   269   275   281   287   293   299   305   311   317   323   329
           335   341   347   353   359   365   371   377   383   389   395   401   407   413
           419   425   431   437   443   449   455   461   467   473   479   485   491   497
           503   509   515   521   527   533   539   545   551   557   563   569   575   581
           587   593   599   605   611   617   623   629   635   641   647   653   659   665
           671   677   683   689   695   701   707   713   719   725   731   737   743   749
           755   761   767   773   779   785   791   797   803   809   815   821   827   833
           839   845   851   857   863   869   875   881   887   893   899   905   911   917
           923   929   935   941   947   953   959   965   971   977   983   989   995  1001
          1007  1013  1019  1025  1031  1037  1043  1049  1055  1061  1067  1073  1079  1083
          1089  1095  1101  1107  1113  1119  1125  1131  1137  1143  1149  1155  1161  1167
          1173  1179  1185  1191  1197  1203  1209  1215  1221  1227  1233  1239  1245  1251
          1257  1263  1269  1275  1281  1287  1293  1299  1305  1311  1317  1323  1329  1335
          1341  1347  1353  1359  1365  1371  1377  1383  1389  1395  1401  1407  1413  1419
          1425  1431  1437  1443  1449  1455  1461  1467  1473  1479  1485  1491  1497  1503
          1509  1515  1521  1527  1533  1539  1545  1551  1557  1563  1569  1575  1581  1587
          1593  1599  1605  1611  1617  1623  1629  1635  1641  1647  1653  1659  1665  1671
          1677  1683  1689  1695  1701  1707  1713  1719  1725  1731  1737  1743  1749  1755
          1761  1767  1773  1779  1785  1791  1797  1803  1809  1815  1821  1827  1833  1839
```

```
SEQ 1157  1845  1851  1857  1863  1869  1875  1881  1887  1893  1899  1905  1911  1917  1923    SEQ 1170
          1929  1935  1941  1947  1953  1959  1965  1971  1977  1983  1989  1995  2001  2007
          2013  2019  2025  2031  2037  2043  2049  2055  2061  2067  2073
.RAND      116  1119  1125  1131  1137  1143  1149  1155  1161  1167  1173  1179  1185  1191
          1197  1203  1209  1215  1221  1227  1233  1239  1245  1251  1257  1263  1269  1275
          1281  1287  1293  1299  1305  1311  1317  1323  1329  1335  1341  1347  1353
.RB        116   121   123   125   127   129   131   133   135   137   139   141   143   145
           147   149   151   161   167   173   179   185   191   197   203   209   215   221
           227   233   239   245   251   257   263   269   275   281   287   293   299   305
           311   317   323   329   335   341   347   353   359   365   371   377   383   389
           395   401   407   413   419   425   431   437   443   449   455   461   467   473
           479   485   491   497   503   509   515   521   527   533   539   545   551   557
           563   569   575   581   587   593   599   605   611   617   623   629   635   641
           647   653   659   665   671   677   683   689   695   701   707   713   719   725
           731   737   743   749   755   761   767   773   779   785   791   797   803   809
           815   821   827   833   839   845   851   857   863   869   875   881   887   893
           899   905   911   917   923   929   935   941   947   953   959   965   971   977
           983   989   995  1001  1007  1013  1019  1025  1031  1037  1043  1049  1055  1061
          1067  1073  1079  1083  1089  1095  1101  1107  1113  1119  1125  1131  1137  1143
          1149  1155  1161  1167  1173  1179  1185  1191  1197  1203  1209  1215  1221  1227
          1233  1239  1245  1251  1257  1263  1269  1275  1281  1287  1293  1299  1305  1311
          1317  1323  1329  1335  1341  1347  1353  1359  1365  1371  1377  1383  1389  1395
          1401  1407  1413  1419  1425  1431  1437  1443  1449  1455  1461  1467  1473  1479
          1485  1491  1497  1503  1509  1515  1521  1527  1533  1539  1545  1551  1557  1563
          1569  1575  1581  1587  1593  1599  1605  1611  1617  1623  1629  1635  1641  1647
          1653  1659  1665  1671  1677  1683  1689  1695  1701  1707  1713  1719  1725  1731
          1737  1743  1749  1755  1761  1767  1773  1779  1785  1791  1797  1803  1809  1815
          1821  1827  1833  1839  1845  1851  1857  1863  1869  1875  1881  1887  1893  1899
          1905  1911  1917  1923  1929  1935  1941  1947  1953  1959  1965  1971  1977  1983
          1989  1995  2001  2007  2013  2019  2025  2031  2037  2043  2049  2055  2061  2067
          2073
.RBAD     2082  2087
.RCCFZ     163   169   175   181   187   193   199   205   211   217   223   229   235   241
           247   253   259   265   271   277   283   289   295   301   307   313   319   325
           331   337   343   349   355   361   367   373   379   385   391   397   403   409
           415   421   427   433   439   445   451   457   463   469   475   481   487   493
           499   505   511   517   523   529   535   541   547   553   559   565   571   577
           583   589   595   601   607   613   619   625   631   637   643   649   655   661
           667   673   679   685   691   697   703   709   715   721   727   733   739   745
           751   757   763   769   775   781   787   793   799   805   811   817   823   829
           835   841   847   853   859   865   871   877   883   889   895   901   907   913
           919   925   931   937   943   949   955   961   967   973   979   985   991   997
          1003  1009  1015  1021  1027  1033  1039  1045  1051  1057  1063  1069  1075  1081
          1085  1091  1097  1103  1109  1115  1121  1127  1133  1139  1145  1151  1157  1163
          1169  1175  1181  1187  1193  1199  1205  1211  1217  1223  1229  1235  1241  1247
          1253  1259  1265  1271  1277  1283  1289  1295  1301  1307  1313  1319  1325  1331
          1337  1343  1349  1355  1361  1367  1373  1379  1385  1391  1397  1403  1409  1415
          1421  1427  1433  1439  1445  1451  1457  1463  1469  1475  1481  1487  1493  1499
          1505  1511  1517  1523  1529  1535  1541  1547  1553  1559  1565  1571  1577  1583
          1589  1595  1601  1607  1613  1619  1625  1631  1637  1643  1649  1655  1661  1667
          1673  1679  1685  1691  1697  1703  1709  1715  1721  1727  1733  1739  1745  1751
          1757  1763  1769  1775  1781  1787  1793  1799  1805  1811  1817  1823  1829  1835
          1841  1847  1853  1859  1865  1871  1877  1883  1889  1895  1901  1907  1913  1919
          1925  1931  1937  1943  1949  1955  1961  1967  1973  1979  1985  1991  1997  2003     SEQ 1171
          2009  2015  2021  2027  2033  2039  2045  2051  2057  2063  2069  2075
.RCENA     163   169   175   181   187   193   199   205   211   217   223   229   235   241
           247   253   259   265   271   277   283   289   295   301   307   313   319   325
           331   337   343   349   355   361   367   373   379   385   391   397   403   409
           415   421   427   433   439   445   451   457   463   469   475   481   487   493
           499   505   511   517   523   529   535   541   547   553   559   565   571   577
           583   589   595   601   607   613   619   625   631   637   643   649   655   661
           667   673   679   685   691   697   703   709   715   721   727   733   739   745
           751   757   763   769   775   781   787   793   799   805   811   817   823   829
           835   841   847   853   859   865   871   877   883   889   895   901   907   913
           919   925   931   937   943   949   955   961   967   973   979   985   991   997
          1003  1009  1015  1021  1027  1033  1039  1045  1051  1057  1063  1069  1075  1081
          1085  1091  1097  1103  1109  1115  1121  1127  1133  1139  1145  1151  1157  1163
          1169  1175  1181  1187  1193  1199  1205  1211  1217  1223  1229  1235  1241  1247
          1253  1259  1265  1271  1277  1283  1289  1295  1301  1307  1313  1319  1325  1331
          1337  1343  1349  1355  1361  1367  1373  1379  1385  1391  1397  1403  1409  1415
          1421  1427  1433  1439  1445  1451  1457  1463  1469  1475  1481  1487  1493  1499
          1505  1511  1517  1523  1529  1535  1541  1547  1553  1559  1565  1571  1577  1583
          1589  1595  1601  1607  1613  1619  1625  1631  1637  1643  1649  1655  1661  1667
          1673  1679  1685  1691  1697  1703  1709  1715  1721  1727  1733  1739  1745  1751
          1757  1763  1769  1775  1781  1787  1793  1799  1805  1811  1817  1823  1829  1835
          1841  1847  1853  1859  1865  1871  1877  1883  1889  1895  1901  1907  1913  1919
          1925  1931  1937  1943  1949  1955  1961  1967  1973  1979  1985  1991  1997  2003
          2009  2015  2021  2027  2033  2039  2045  2051  2057  2063  2069  2075
.RCJV      163   169   175   181   187   193   199   205   211   217   223   229   235   241
           247   253   259   265   271   277   283   289   295   301   307   313   319   325
           331   337   343   349   355   361   367   373   379   385   391   397   403   409
           415   421   427   433   439   445   451   457   463   469   475   481   487   493
           499   505   511   517   523   529   535   541   547   553   559   565   571   577
           583   589   595   601   607   613   619   625   631   637   643   649   655   661
           667   673   679   685   691   697   703   709   715   721   727   733   739   745
           751   757   763   769   775   781   787   793   799   805   811   817   823   829
           835   841   847   853   859   865   871   877   883   889   895   901   907   913
           919   925   931   937   943   949   955   961   967   973   979   985   991   997
          1003  1009  1015  1021  1027  1033  1039  1045  1051  1057  1063  1069  1075  1081
          1085  1091  1097  1103  1109  1115  1121  1127  1133  1139  1145  1151  1157  1163
          1169  1175  1181  1187  1193  1199  1205  1211  1217  1223  1229  1235  1241  1247
          1253  1259  1265  1271  1277  1283  1289  1295  1301  1307  1313  1319  1325  1331
          1337  1343  1349  1355  1361  1367  1373  1379  1385  1391  1397  1403  1409  1415
          1421  1427  1433  1439  1445  1451  1457  1463  1469  1475  1481  1487  1493  1499
          1505  1511  1517  1523  1529  1535  1541  1547  1553  1559  1565  1571  1577  1583
          1589  1595  1601  1607  1613  1619  1625  1631  1637  1643  1649  1655  1661  1667
          1673  1679  1685  1691  1697  1703  1709  1715  1721  1727  1733  1739  1745  1751
          1757  1763  1769  1775  1781  1787  1793  1799  1805  1811  1817  1823  1829  1835
          1841  1847  1853  1859  1865  1871  1877  1883  1889  1895  1901  1907  1913  1919
          1925  1931  1937  1943  1949  1955  1961  1967  1973  1979  1985  1991  1997  2003
          2009  2015  2021  2027  2033  2039  2045  2051  2057  2063  2069  2075
.RCONT     121   123   125   127   129   131   133   135   137   139   141   143   145   147
           149   151   161   167   173   179   185   191   197   203   209   215   221   227
           233   239   245   251   257   263   269   275   281   287   293   299   305   311
           317   323   329   335   341   347   353   359   365   371   377   383   389   395
           401   407   413   419   425   431   437   443   449   455   461   467   473   479
```

```
1159
          485   491   497   503   509   515   521   527   533   539   545   551   557   563     SEQ 1172
          569   575   581   587   593   599   605   611   617   623   629   635   641   647
          653   659   665   671   677   683   689   695   701   707   713   719   725   731
          737   743   749   755   761   767   773   779   785   791   797   803   809   815
          821   827   833   839   845   851   857   863   869   875   881   887   893   899
          905   911   917   923   929   935   941   947   953   959   965   971   977   983
          989   995  1001  1007  1013  1019  1025  1031  1037  1043  1049  1055  1061  1067
         1073  1079  1083  1089  1095  1101  1107  1113  1119  1125  1131  1137  1143  1149
         1155  1161  1167  1173  1179  1185  1191  1197  1203  1209  1215  1221  1227  1233
         1239  1245  1251  1257  1263  1269  1275  1281  1287  1293  1299  1305  1311  1317
         1323  1329  1335  1341  1347  1353  1359  1365  1371  1377  1383  1389  1395  1401
         1407  1413  1419  1425  1431  1437  1443  1449  1455  1461  1467  1473  1479  1485
         1491  1497  1503  1509  1515  1521  1527  1533  1539  1545  1551  1557  1563  1569
         1575  1581  1587  1593  1599  1605  1611  1617  1623  1629  1635  1641  1647  1653
         1659  1665  1671  1677  1683  1689  1695  1701  1707  1713  1719  1725  1731  1737
         1743  1749  1755  1761  1767  1773  1779  1785  1791  1797  1803  1809  1815  1821
         1827  1833  1839  1845  1851  1857  1863  1869  1875  1881  1887  1893  1899  1905
         1911  1917  1923  1929  1935  1941  1947  1953  1959  1965  1971  1977  1983  1989
         1995  2001  2007  2013  2019  2025  2031  2037  2043  2049  2055  2061  2067  2073
.RD       116   121   123   125   127   129   131   133   135   137   139   141   143   145
          147   149   151   153   161   163   167   169   173   175   179   181   185   187
          191   193   197   199   203   205   209   211   215   217   221   223   227   229
          233   235   239   241   245   247   251   253   257   259   263   265   269   271
          275   277   281   283   287   289   293   295   299   301   305   307   311   313
          317   319   323   325   329   331   335   337   341   343   347   349   353   355
          359   361   365   367   371   373   377   379   383   385   389   391   395   397
          401   403   407   409   413   415   419   421   425   427   431   433   437   439
          443   445   449   451   455   457   461   463   467   469   473   475   479   481
          485   487   491   493   497   499   503   505   509   511   515   517   521   523
          527   529   533   535   539   541   545   547   551   553   557   559   563   565
          569   571   575   577   581   583   587   589   593   595   599   601   605   607
          611   613   617   619   623   625   629   631   635   637   641   643   647   649
          653   655   659   661   665   667   671   673   677   679   683   685   689   691
          695   697   701   703   707   709   713   715   719   721   725   727   731   733
          737   739   743   745   749   751   755   757   761   763   767   769   773   775
          779   781   785   787   791   793   797   799   803   805   809   811   815   817
          821   823   827   829   833   835   839   841   845   847   851   853   857   859
          863   865   869   871   875   877   881   883   887   889   893   895   899   901
          905   907   911   913   917   919   923   925   929   931   935   937   941   943
          947   949   953   955   959   961   965   967   971   973   977   979   983   985
          989   991   995   997  1001  1003  1007  1009  1013  1015  1019  1021  1025  1027
         1031  1033  1037  1039  1043  1045  1049  1051  1055  1057  1061  1063  1067  1069
         1073  1075  1079  1081  1083  1085  1089  1091  1095  1097  1101  1103  1107  1109
         1113  1115  1119  1121  1125  1127  1131  1133  1137  1139  1143  1145  1149  1151
         1155  1157  1161  1163  1167  1169  1173  1175  1179  1181  1185  1187  1191  1193
         1197  1199  1203  1205  1209  1211  1215  1217  1221  1223  1227  1229  1233  1235
         1239  1241  1245  1247  1251  1253  1257  1259  1263  1265  1269  1271  1275  1277
         1281  1283  1287  1289  1293  1295  1299  1301  1305  1307  1311  1313  1317  1319
         1323  1325  1329  1331  1335  1337  1341  1343  1347  1349  1353  1355  1359  1361
         1365  1367  1371  1373  1377  1379  1383  1385  1389  1391  1395  1397  1401  1403
         1407  1409  1413  1415  1419  1421  1425  1427  1431  1433  1437  1439  1443  1445
         1449  1451  1455  1457  1461  1463  1467  1469  1473  1475  1479  1481  1485  1487
         1491  1493  1497  1499  1503  1505  1509  1511  1515  1517  1521  1523  1527  1529

         1533  1535  1539  1541  1545  1547  1551  1553  1557  1559  1563  1565  1569  1571
```

```
1160    1533  1535  1539  1541  1545  1547  1551  1553  1557  1559  1563  1565  1569  1571     SEQ 1173   DFP
        1575  1577  1581  1583  1587  1589  1593  1595  1599  1601  1605  1607  1611  1613                DFP
        1617  1619  1623  1625  1629  1631  1635  1637  1641  1643  1647  1649  1653  1655
        1659  1661  1665  1667  1671  1673  1677  1679  1683  1685  1689  1691  1695  1697
        1701  1703  1707  1709  1713  1715  1719  1721  1725  1727  1731  1733  1737  1739
        1743  1745  1749  1751  1755  1757  1761  1763  1767  1769  1773  1775  1779  1781
        1785  1787  1791  1793  1797  1799  1803  1805  1809  1811  1815  1817  1821  1823
        1827  1829  1833  1835  1839  1841  1845  1847  1851  1853  1857  1859  1863  1865
        1869  1871  1875  1877  1881  1883  1887  1889  1893  1895  1899  1901  1905  1907
        1911  1913  1917  1919  1923  1925  1929  1931  1935  1937  1941  1943  1947  1949
        1953  1955  1959  1961  1965  1967  1971  1973  1977  1979  1983  1985  1989  1991
        1995  1997  2001  2003  2007  2009  2013  2015  2019  2021  2025  2027  2031  2033
        2037  2039  2043  2045  2049  2051  2055  2057  2061  2063  2067  2069  2073  2075
.RJ      116   153   163   165   169   171   175   177   181   183   187   189   193   195
         199   201   205   207   211   213   217   219   223   225   229   231   235   237
         241   243   247   249   253   255   259   261   265   267   271   273   277   279
         283   285   289   291   295   297   301   303   307   309   313   315   319   321
         325   327   331   333   337   339   343   345   349   351   355   357   361   363
         367   369   373   375   379   381   385   387   391   393   397   399   403   405
         409   411   415   417   421   423   427   429   433   435   439   441   445   447
         451   453   457   459   463   465   469   471   475   477   481   483   487   489
         493   495   499   501   505   507   511   513   517   519   523   525   529   531
         535   537   541   543   547   549   553   555   559   561   565   567   571   573
         577   579   583   585   589   591   595   597   601   603   607   609   613   615
         619   621   625   627   631   633   637   639   643   645   649   651   655   657
         661   663   667   669   673   675   679   681   685   687   691   693   697   699
         703   705   709   711   715   717   721   723   727   729   733   735   739   741
         745   747   751   753   757   759   763   765   769   771   775   777   781   783
         787   789   793   795   799   801   805   807   811   813   817   819   823   825
         829   831   835   837   841   843   847   849   853   855   859   861   865   867
         871   873   877   879   883   885   889   891   895   897   901   903   907   909
         913   915   919   921   925   927   931   933   937   939   943   945   949   951
         955   957   961   963   967   969   973   975   979   981   985   987   991   993
         997   999  1003  1005  1009  1011  1015  1017  1021  1023  1027  1029  1033  1035
        1039  1041  1045  1047  1051  1053  1057  1059  1063  1065  1069  1071  1075  1077
        1081  1085  1087  1091  1093  1097  1099  1103  1105  1109  1111  1115  1117  1121
        1123  1127  1129  1133  1135  1139  1141  1145  1147  1151  1153  1157  1159  1163
        1165  1169  1171  1175  1177  1181  1183  1187  1189  1193  1195  1199  1201  1205
        1207  1211  1213  1217  1219  1223  1225  1229  1231  1235  1237  1241  1243  1247
        1249  1253  1255  1259  1261  1265  1267  1271  1273  1277  1279  1283  1285  1289
        1291  1295  1297  1301  1303  1307  1309  1313  1315  1319  1321  1325  1327  1331
        1333  1337  1339  1343  1345  1349  1351  1355  1357  1361  1363  1367  1369  1373
        1375  1379  1381  1385  1387  1391  1393  1397  1399  1403  1405  1409  1411  1415
        1417  1421  1423  1427  1429  1433  1435  1439  1441  1445  1447  1451  1453  1457
        1459  1463  1465  1469  1471  1475  1477  1481  1483  1487  1489  1493  1495  1499
        1501  1505  1507  1511  1513  1517  1519  1523  1525  1529  1531  1535  1537  1541
        1543  1547  1549  1553  1555  1559  1561  1565  1567  1571  1573  1577  1579  1583
        1585  1589  1591  1595  1597  1601  1603  1607  1609  1613  1615  1619  1621  1625
        1627  1631  1633  1637  1639  1643  1645  1649  1651  1655  1657  1661  1663  1667
        1669  1673  1675  1679  1681  1685  1687  1691  1693  1697  1699  1703  1705  1709
        1711  1715  1717  1721  1723  1727  1729  1733  1735  1739  1741  1745  1747  1751
        1753  1757  1759  1763  1765  1769  1771  1775  1777  1781  1783  1787  1789  1793
        1795  1799  1801  1805  1807  1811  1813  1817  1819  1823  1825  1829  1831  1835
```

```
        1837  1841  1843  1847  1849  1853  1855  1859  1861  1865  1867  1871  1873  1877
```

```
1161           1837  1841  1843  1847  1849  1853  1855  1859  1861  1865  1867  1871  1873  1877     SEQ 1174
               1879  1883  1885  1889  1891  1895  1897  1901  1903  1907  1909  1913  1915  1919
               1921  1925  1927  1931  1933  1937  1939  1943  1945  1949  1951  1955  1957  1961
               1963  1967  1969  1973  1975  1979  1981  1985  1987  1991  1993  1997  1999  2003
               2005  2009  2011  2015  2017  2021  2023  2027  2029  2033  2035  2039  2041  2045
               2047  2051  2053  2057  2059  2063  2065  2069  2071  2075  2077  2082  2087
     .RJMAP     116   153   165   171   177   183   189   195   201   207   213   219   225   231
                237   243   249   255   261   267   273   279   285   291   297   303   309   315
                321   327   333   339   345   351   357   363   369   375   381   387   393   399
                405   411   417   423   429   435   441   447   453   459   465   471   477   483
                489   495   501   507   513   519   525   531   537   543   549   555   561   567
                573   579   585   591   597   603   609   615   621   627   633   639   645   651
                657   663   669   675   681   687   693   699   705   711   717   723   729   735
                741   747   753   759   765   771   777   783   789   795   801   807   813   819
                825   831   837   843   849   855   861   867   873   879   885   891   897   903
                909   915   921   927   933   939   945   951   957   963   969   975   981   987
                993   999  1005  1011  1017  1023  1029  1035  1041  1047  1053  1059  1065  1071
               1077  1087  1093  1099  1105  1111  1117  1123  1129  1135  1141  1147  1153  1159
               1165  1171  1177  1183  1189  1195  1201  1207  1213  1219  1225  1231  1237  1243
               1249  1255  1261  1267  1273  1279  1285  1291  1297  1303  1309  1315  1321  1327
               1333  1339  1345  1351  1357  1363  1369  1375  1381  1387  1393  1399  1405  1411
               1417  1423  1429  1435  1441  1447  1453  1459  1465  1471  1477  1483  1489  1495
               1501  1507  1513  1519  1525  1531  1537  1543  1549  1555  1561  1567  1573  1579
               1585  1591  1597  1603  1609  1615  1621  1627  1633  1639  1645  1651  1657  1663
               1669  1675  1681  1687  1693  1699  1705  1711  1717  1723  1729  1735  1741  1747
               1753  1759  1765  1771  1777  1783  1789  1795  1801  1807  1813  1819  1825  1831
               1837  1843  1849  1855  1861  1867  1873  1879  1885  1891  1897  1903  1909  1915
               1921  1927  1933  1939  1945  1951  1957  1963  1969  1975  1981  1987  1993  1999
               2005  2011  2017  2023  2029  2035  2041  2047  2053  2059  2065  2071  2077  2082
               2087
     .RMGC      121   123   125   127   129   131   133   135   137   139   141   143   145   147
                149   151
     .RNAND    1359  1365  1371  1377  1383  1389  1395  1401  1407  1413  1419  1425  1431  1437
               1443  1449  1455  1461  1467  1473  1479  1485  1491  1497  1503  1509  1515  1521
               1527  1533  1539  1545  1551  1557  1563  1569  1575  1581  1587  1593
     .ROR       121   123   125   127   129   131   133   135   137   139   141   143   145   147
                149   151   163   169   175   181   187   193   199   205   211   217   223   229
                235   241   247   253   259   265   271   277   283   289   295   301   307   313
                319   325   331   337   343   349   355   361   367   373   379   385   391   397
                403   409   415   421   427   433   439   445   451   457   463   469   475   481
                487   493   499   505   511   517   523   529   535   541   547   553   559   565
                571   577   583   589   595   601   607   613   619   625   631   637   643   649
                655   661   667   673   679   685   691   697   703   709   715   721   727   733
                739   745   751   757   763   769   775   781   787   793   799   805   811   817
                823   829   835   841   847   853   859   865   871   877   881   883   887   889
                893   895   899   901   905   907   911   913   917   919   923   925   929   931
                935   937   941   943   947   949   953   955   959   961   965   967   971   973
                977   979   983   985   989   991   995   997  1001  1003  1007  1009  1013  1015
               1019  1021  1025  1027  1031  1033  1037  1039  1043  1045  1049  1051  1055  1057
               1061  1063  1067  1069  1073  1075  1079  1081  1083  1085  1089  1091  1095  1097
               1101  1103  1107  1109  1113  1115  1121  1127  1133  1139  1145  1151  1157  1163
               1169  1175  1181  1187  1193  1199  1205  1211  1217  1223  1229  1235  1241  1247
               1253  1259  1265  1271  1277  1283  1289  1295  1301  1307  1313  1319  1325  1331


               1337  1343  1349  1355  1361  1367  1373  1379  1385  1391  1397  1403  1409  1415
```

```
             1337   1343   1349   1355   1361   1367   1373   1379   1385   1391   1397   1403   1409   1415
             1421   1427   1433   1439   1445   1451   1457   1463   1469   1475   1481   1487   1493   1499
             1505   1511   1517   1523   1529   1535   1541   1547   1553   1559   1565   1571   1577   1583
             1589   1595   1601   1607   1613   1619   1625   1631   1637   1643   1649   1655   1661   1667
             1673   1679   1685   1691   1697   1703   1709   1715   1721   1727   1733   1739   1745   1751
             1757   1763   1769   1775   1781   1787   1793   1799   1805   1811   1817   1823   1829   1835
             1841   1847   1853   1859   1865   1871   1877   1883   1889   1895   1901   1907   1913   1919
             1925   1931   1937   1943   1949   1955   1961   1967   1973   1979   1985   1991   1997   2003
             2009   2015   2021   2027   2033   2039   2045   2051   2057   2063   2069   2075
.RPLUS        161    167    173    179    185    191    197    203    209    215    221    227    233    239
              245    251    257    263    269    275    281    287    293    299    305    311    317    323
              329    335    341    347    353    359    365    371    377    383    389    395
.RRMIN        641    647    653    659    665    671    677    683    689    695    701    707    713    719
              725    731    737    743    749    755    761    767    773    779    785    791    797    803
              809    815    821    827    833    839    845    851    857    863    869    875
.RSOA         281    287    293    299    305    521    527    533    539    545    761    767    773    779
              785   1001   1007   1013   1019   1025   1239   1245   1251   1257   1263   1479   1485   1491
             1497   1503   1719   1725   1731   1737   1743   1959   1965   1971   1977   1983
.RSOB         251    257    263    269    275    491    497    503    509    515    731    737    743    749
              755    971    977    983    989    995   1209   1215   1221   1227   1233   1449   1455   1461
             1467   1473   1689   1695   1701   1707   1713   1929   1935   1941   1947   1953
.RSOQ         163    169    175    181    187    193    199    205    211    217    221    223    227    229
              233    235    239    241    245    247    253    259    265    271    277    283    289    295
              301    307    313    319    325    331    337    343    349    355    361    367    373    379
              385    391    397    403    409    415    421    427    433    439    445    451    457    461
              463    467    469    473    475    479    481    485    487    493    499    505    511    517
              523    529    535    541    547    553    559    565    571    577    583    589    595    601
              607    613    619    625    631    637    643    649    655    661    667    673    679    685
              691    697    701    703    707    709    713    715    719    721    725    727    733    739
              745    751    757    763    769    775    781    787    793    799    805    811    817    823
              829    835    841    847    853    859    865    871    877    883    889    895    901    907
              913    919    925    931    937    941    943    947    949    953    955    959    961    965
              967    973    979    985    991    997   1003   1009   1015   1021   1027   1033   1039   1045
             1051   1057   1063   1069   1075   1081   1085   1091   1097   1103   1109   1115   1121   1127
             1133   1139   1145   1151   1157   1163   1169   1175   1179   1181   1185   1187   1191   1193
             1197   1199   1203   1205   1211   1217   1223   1229   1235   1241   1247   1253   1259   1265
             1271   1277   1283   1289   1295   1301   1307   1313   1319   1325   1331   1337   1343   1349
             1355   1361   1367   1373   1379   1385   1391   1397   1403   1409   1415   1419   1421   1425
             1427   1431   1433   1437   1439   1443   1445   1451   1457   1463   1469   1475   1481   1487
             1493   1499   1505   1511   1517   1523   1529   1535   1541   1547   1553   1559   1565   1571
             1577   1583   1589   1595   1601   1607   1613   1619   1625   1631   1637   1643   1649   1655
             1659   1661   1665   1667   1671   1673   1677   1679   1683   1685   1691   1697   1703   1709
             1715   1721   1727   1733   1739   1745   1751   1757   1763   1769   1775   1781   1787   1793
             1799   1805   1811   1817   1823   1829   1835   1841   1847   1853   1859   1865   1871   1877
             1883   1889   1895   1899   1901   1905   1907   1911   1913   1917   1919   1923   1925   1931
             1937   1943   1949   1955   1961   1967   1973   1979   1985   1991   1997   2003   2009   2015
             2021   2027   2033   2039   2045   2051   2057   2063   2069   2075
.RSAB         191    197    203    209    215    431    437    443    449    455    671    677    683    689
              695    911    917    923    929    935   1149   1155   1161   1167   1173   1389   1395   1401
             1407   1413   1629   1635   1641   1647   1653   1869   1875   1881   1887   1893
.RSAQ         161    167    173    179    185    401    407    413    419    425    641    647    653    659
              665    881    887    893    899    905   1119   1125   1131   1137   1143   1359   1365   1371
             1377   1383   1599   1605   1611   1617   1623   1839   1845   1851   1857   1863
```

```
.RSDO     116   121   123   125   127   129   131   133   135   137   139   141   143   145
          147   149   151   371   377   383   389   395   611   617   623   629   635   851
          857   863   869   875  1089  1095  1101  1107  1113  1329  1335  1341  1347  1353
         1569  1575  1581  1587  1593  1809  1815  1821  1827  1833  2049  2055  2061  2067
         2073
.RSDA     311   317   323   329   335   551   557   563   569   575   791   797   803   809
          815  1031  1037  1043  1049  1055  1269  1275  1281  1287  1293  1509  1515  1521
         1527  1533  1749  1755  1761  1767  1773  1989  1995  2001  2007  2013
.RSDQ     341   347   353   359   365   581   587   593   599   605   821   827   833   839
          845  1061  1067  1073  1079  1083  1299  1305  1311  1317  1323  1539  1545  1551
         1557  1563  1779  1785  1791  1797  1803  2019  2025  2031  2037  2043
.RSKCN    121   123   125   127   129   131   133   135   137   139   141   143   145   147
          149   151
.RSMIN    401   407   413   419   425   431   437   443   449   455   461   467   473   479
          485   491   497   503   509   515   521   527   533   539   545   551   557   563
          569   575   581   587   593   599   605   611   617   623   629   635
.RXNOR   1839  1845  1851  1857  1863  1869  1875  1881  1887  1893  1899  1905  1911  1917
         1923  1929  1935  1941  1947  1953  1959  1965  1971  1977  1983  1989  1995  2001
         2007  2013  2019  2025  2031  2037  2043  2049  2055  2061  2067  2073
.RXOR    1599  1605  1611  1617  1623  1629  1635  1641  1647  1653  1659  1665  1671  1677
         1683  1689  1695  1701  1707  1713  1719  1725  1731  1737  1743  1749  1755  1761
         1767  1773  1779  1785  1791  1797  1803  1809  1815  1821  1827  1833
```

```
CALC   116  121  123  125  127  129  131  133  135  137  139  141  143  145     SEQ 1177
       147  149  151  153  161  163  165  167  169  171  173  175  177  179
       181  183  185  187  189  191  193  195  197  199  201  203  205  207
       209  211  213  215  217  219  221  223  225  227  229  231  233  235
       237  239  241  243  245  247  249  251  253  255  257  259  261  263
       265  267  269  271  273  275  277  279  281  283  285  287  289  291
       293  295  297  299  301  303  305  307  309  311  313  315  317  319
       321  323  325  327  329  331  333  335  337  339  341  343  345  347
       349  351  353  355  357  359  361  363  365  367  369  371  373  375
       377  379  381  383  385  387  389  391  393  395  397  399  401  403
       405  407  409  411  413  415  417  419  421  423  425  427  429  431
       433  435  437  439  441  443  445  447  449  451  453  455  457  459
       461  463  465  467  469  471  473  475  477  479  481  483  485  487
       489  491  493  495  497  499  501  503  505  507  509  511  513  515
       517  519  521  523  525  527  529  531  533  535  537  539  541  543
       545  547  549  551  553  555  557  559  561  563  565  567  569  571
       573  575  577  579  581  583  585  587  589  591  593  595  597  599
       601  603  605  607  609  611  613  615  617  619  621  623  625  627
       629  631  633  635  637  639  641  643  645  647  649  651  653  655
       657  659  661  663  665  667  669  671  673  675  677  679  681  683
       685  687  689  691  693  695  697  699  701  703  705  707  709  711
       713  715  717  719  721  723  725  727  729  731  733  735  737  739
       741  743  745  747  749  751  753  755  757  759  761  763  765  767
       769  771  773  775  777  779  781  783  785  787  789  791  793  795
       797  799  801  803  805  807  809  811  813  815  817  819  821  823
       825  827  829  831  833  835  837  839  841  843  845  847  849  851
       853  855  857  859  861  863  865  867  869  871  873  875  877  879
       881  883  885  887  889  891  893  895  897  899  901  903  905  907
       909  911  913  915  917  919  921  923  925  927  929  931  933  935
       937  939  941  943  945  947  949  951  953  955  957  959  961  963
       965  967  969  971  973  975  977  979  981  983  985  987  989  991
       993  995  997  999 1001 1003 1005 1007 1009 1011 1013 1015 1017 1019
      1021 1023 1025 1027 1029 1031 1033 1035 1037 1039 1041 1043 1045 1047
      1049 1051 1053 1055 1057 1059 1061 1063 1065 1067 1069 1071 1073 1075
      1077 1079 1081 1083 1085 1087 1089 1091 1093 1095 1097 1099 1101 1103
      1105 1107 1109 1111 1113 1115 1117 1119 1121 1123 1125 1127 1129 1131
      1133 1135 1137 1139 1141 1143 1145 1147 1149 1151 1153 1155 1157 1159
      1161 1163 1165 1167 1169 1171 1173 1175 1177 1179 1181 1183 1185 1187
      1189 1191 1193 1195 1197 1199 1201 1203 1205 1207 1209 1211 1213 1215
      1217 1219 1221 1223 1225 1227 1229 1231 1233 1235 1237 1239 1241 1243
      1245 1247 1249 1251 1253 1255 1257 1259 1261 1263 1265 1267 1269 1271
      1273 1275 1277 1279 1281 1283 1285 1287 1289 1291 1293 1295 1297 1299
      1301 1303 1305 1307 1309 1311 1313 1315 1317 1319 1321 1323 1325 1327
      1329 1331 1333 1335 1337 1339 1341 1343 1345 1347 1349 1351 1353 1355
      1357 1359 1361 1363 1365 1367 1369 1371 1373 1375 1377 1379 1381 1383
      1385 1387 1389 1391 1393 1395 1397 1399 1401 1403 1405 1407 1409 1411
      1413 1415 1417 1419 1421 1423 1425 1427 1429 1431 1433 1435 1437 1439
      1441 1443 1445 1447 1449 1451 1453 1455 1457 1459 1461 1463 1465 1467
      1469 1471 1473 1475 1477 1479 1481 1483 1485 1487 1489 1491 1493 1495
      1497 1499 1501 1503 1505 1507 1509 1511 1513 1515 1517 1519 1521 1523
      1525 1527 1529 1531 1533 1535 1537 1539 1541 1543 1545 1547 1549 1551
      1553 1555 1557 1559 1561 1563 1565 1567 1569 1571 1573 1575 1577 1579
      1581 1583 1585 1587 1589 1591 1593 1595 1597 1599 1601 1603 1605 1607
```

```
Q 1165    1609  1611  1613  1615  1617  1619  1621  1623  1625  1627  1629  1631  1633  1635    SEQ 1178
          1637  1639  1641  1643  1645  1647  1649  1651  1653  1655  1657  1659  1661  1663
          1665  1667  1669  1671  1673  1675  1677  1679  1681  1683  1685  1687  1689  1691
          1693  1695  1697  1699  1701  1703  1705  1707  1709  1711  1713  1715  1717  1719
          1721  1723  1725  1727  1729  1731  1733  1735  1737  1739  1741  1743  1745  1747
          1749  1751  1753  1755  1757  1759  1761  1763  1765  1767  1769  1771  1773  1775
          1777  1779  1781  1783  1785  1787  1789  1791  1793  1795  1797  1799  1801  1803
          1805  1807  1809  1811  1813  1815  1817  1819  1821  1823  1825  1827  1829  1831
          1833  1835  1837  1839  1841  1843  1845  1847  1849  1851  1853  1855  1857  1859
          1861  1863  1865  1867  1869  1871  1873  1875  1877  1879  1881  1883  1885  1887
          1889  1891  1893  1895  1897  1899  1901  1903  1905  1907  1909  1911  1913  1915
          1917  1919  1921  1923  1925  1927  1929  1931  1933  1935  1937  1939  1941  1943
          1945  1947  1949  1951  1953  1955  1957  1959  1961  1963  1965  1967  1969  1971
          1973  1975  1977  1979  1981  1983  1985  1987  1989  1991  1993  1995  1997  1999
          2001  2003  2005  2007  2009  2011  2013  2015  2017  2019  2021  2023  2025  2027
          2029  2031  2033  2035  2037  2039  2041  2043  2045  2047  2049  2051  2053  2055
          2057  2059  2061  2063  2065  2067  2069  2071  2073  2075  2077  2082  2087
CONCAT     116   121   123   125   127   129   131   133   135   137   139   141   143   145
           147   149   151   153   161   163   165   167   169   171   173   175   177   179
           181   183   185   187   189   191   193   195   197   199   201   203   205   207
           209   211   213   215   217   219   221   223   225   227   229   231   233   235
           237   239   241   243   245   247   249   251   253   255   257   259   261   263
           265   267   269   271   273   275   277   279   281   283   285   287   289   291
           293   295   297   299   301   303   305   307   309   311   313   315   317   319
           321   323   325   327   329   331   333   335   337   339   341   343   345   347
           349   351   353   355   357   359   361   363   365   367   369   371   373   375
           377   379   381   383   385   387   389   391   393   395   397   399   401   403
           405   407   409   411   413   415   417   419   421   423   425   427   429   431
           433   435   437   439   441   443   445   447   449   451   453   455   457   459
           461   463   465   467   469   471   473   475   477   479   481   483   485   487
           489   491   493   495   497   499   501   503   505   507   509   511   513   515
           517   519   521   523   525   527   529   531   533   535   537   539   541   543
           545   547   549   551   553   555   557   559   561   563   565   567   569   571
           573   575   577   579   581   583   585   587   589   591   593   595   597   599
           601   603   605   607   609   611   613   615   617   619   621   623   625   627
           629   631   633   635   637   639   641   643   645   647   649   651   653   655
           657   659   661   663   665   667   669   671   673   675   677   679   681   683
           685   687   689   691   693   695   697   699   701   703   705   707   709   711
           713   715   717   719   721   723   725   727   729   731   733   735   737   739
           741   743   745   747   749   751   753   755   757   759   761   763   765   767
           769   771   773   775   777   779   781   783   785   787   789   791   793   795
           797   799   801   803   805   807   809   811   813   815   817   819   821   823
           825   827   829   831   833   835   837   839   841   843   845   847   849   851
           853   855   857   859   861   863   865   867   869   871   873   875   877   879
           881   883   885   887   889   891   893   895   897   899   901   903   905   907
           909   911   913   915   917   919   921   923   925   927   929   931   933   935
           937   939   941   943   945   947   949   951   953   955   957   959   961   963
           965   967   969   971   973   975   977   979   981   983   985   987   989   991
           993   995   997   999  1001  1003  1005  1007  1009  1011  1013  1015  1017  1019
          1021  1023  1025  1027  1029  1031  1033  1035  1037  1039  1041  1043  1045  1047
          1049  1051  1053  1055  1057  1059  1061  1063  1065  1067  1069  1071  1073  1075
          1077  1079  1081  1083  1085  1087  1089  1091  1093  1095  1097  1099  1101  1103
          1105  1107  1109  1111  1113  1115  1117  1119  1121  1123  1125  1127  1129  1131
```

EQ 1166

SEQ 1179

	1133	1135	1137	1139	1141	1143	1145	1147	1149	1151	1153	1155	1157	1159
	1161	1163	1165	1167	1169	1171	1173	1175	1177	1179	1181	1183	1185	1187
	1189	1191	1193	1195	1197	1199	1201	1203	1205	1207	1209	1211	1213	1215
	1217	1219	1221	1223	1225	1227	1229	1231	1233	1235	1237	1239	1241	1243
	1245	1247	1249	1251	1253	1255	1257	1259	1261	1263	1265	1267	1269	1271
	1273	1275	1277	1279	1281	1283	1285	1287	1289	1291	1293	1295	1297	1299
	1301	1303	1305	1307	1309	1311	1313	1315	1317	1319	1321	1323	1325	1327
	1329	1331	1333	1335	1337	1339	1341	1343	1345	1347	1349	1351	1353	1355
	1357	1359	1361	1363	1365	1367	1369	1371	1373	1375	1377	1379	1381	1383
	1385	1387	1389	1391	1393	1395	1397	1399	1401	1403	1405	1407	1409	1411
	1413	1415	1417	1419	1421	1423	1425	1427	1429	1431	1433	1435	1437	1439
	1441	1443	1445	1447	1449	1451	1453	1455	1457	1459	1461	1463	1465	1467
	1469	1471	1473	1475	1477	1479	1481	1483	1485	1487	1489	1491	1493	1495
	1497	1499	1501	1503	1505	1507	1509	1511	1513	1515	1517	1519	1521	1523
	1525	1527	1529	1531	1533	1535	1537	1539	1541	1543	1545	1547	1549	1551
	1553	1555	1557	1559	1561	1563	1565	1567	1569	1571	1573	1575	1577	1579
	1581	1583	1585	1587	1589	1591	1593	1595	1597	1599	1601	1603	1605	1607
	1609	1611	1613	1615	1617	1619	1621	1623	1625	1627	1629	1631	1633	1635
	1637	1639	1641	1643	1645	1647	1649	1651	1653	1655	1657	1659	1661	1663
	1665	1667	1669	1671	1673	1675	1677	1679	1681	1683	1685	1687	1689	1691
	1693	1695	1697	1699	1701	1703	1705	1707	1709	1711	1713	1715	1717	1719
	1721	1723	1725	1727	1729	1731	1733	1735	1737	1739	1741	1743	1745	1747
	1749	1751	1753	1755	1757	1759	1761	1763	1765	1767	1769	1771	1773	1775
	1777	1779	1781	1783	1785	1787	1789	1791	1793	1795	1797	1799	1801	1803
	1805	1807	1809	1811	1813	1815	1817	1819	1821	1823	1825	1827	1829	1831
	1833	1835	1837	1839	1841	1843	1845	1847	1849	1851	1853	1855	1857	1859
	1861	1863	1865	1867	1869	1871	1873	1875	1877	1879	1881	1883	1885	1887
	1889	1891	1893	1895	1897	1899	1901	1903	1905	1907	1909	1911	1913	1915
	1917	1919	1921	1923	1925	1927	1929	1931	1933	1935	1937	1939	1941	1943
	1945	1947	1949	1951	1953	1955	1957	1959	1961	1963	1965	1967	1969	1971
	1973	1975	1977	1979	1981	1983	1985	1987	1989	1991	1993	1995	1997	1999
	2001	2003	2005	2007	2009	2011	2013	2015	2017	2019	2021	2023	2025	2027
	2029	2031	2033	2035	2037	2039	2041	2043	2045	2047	2049	2051	2053	2055
	2057	2059	2061	2063	2065	2067	2069	2071	2073	2075	2077	2082	2087	
FIELD	116	121	123	125	127	129	131	133	135	137	139	141	143	145
	147	149	151	153	161	163	165	167	169	171	173	175	177	179
	181	183	185	187	189	191	193	195	197	199	201	203	205	207
	209	211	213	215	217	219	221	223	225	227	229	231	233	235
	237	239	241	243	245	247	249	251	253	255	257	259	261	263
	265	267	269	271	273	275	277	279	281	283	285	287	289	291
	293	295	297	299	301	303	305	307	309	311	313	315	317	319
	321	323	325	327	329	331	333	335	337	339	341	343	345	347
	349	351	353	355	357	359	361	363	365	367	369	371	373	375
	377	379	381	383	385	387	389	391	393	395	397	399	401	403
	405	407	409	411	413	415	417	419	421	423	425	427	429	431
	433	435	437	439	441	443	445	447	449	451	453	455	457	459
	461	463	465	467	469	471	473	475	477	479	481	483	485	487
	489	491	493	495	497	499	501	503	505	507	509	511	513	515
	517	519	521	523	525	527	529	531	533	535	537	539	541	543
	545	547	549	551	553	555	557	559	561	563	565	567	569	571
	573	575	577	579	581	583	585	587	589	591	593	595	597	599
	601	603	605	607	609	611	613	615	617	619	621	623	625	627
	629	631	633	635	637	639	641	643	645	647	649	651	653	655

SEQ 1167

	657	659	661	663	665	667	669	671	673	675	677	679	681	683
	685	687	689	691	693	695	697	699	701	703	705	707	709	711
	713	715	717	719	721	723	725	727	729	731	733	735	737	739
	741	743	745	747	749	751	753	755	757	759	761	763	765	767
	769	771	773	775	777	779	781	783	785	787	789	791	793	795
	797	799	801	803	805	807	809	811	813	815	817	819	821	823
	825	827	829	831	833	835	837	839	841	843	845	847	849	851
	853	855	857	859	861	863	865	867	869	871	873	875	877	879
	881	883	885	887	889	891	893	895	897	899	901	903	905	907
	909	911	913	915	917	919	921	923	925	927	929	931	933	935
	937	939	941	943	945	947	949	951	953	955	957	959	961	963
	965	967	969	971	973	975	977	979	981	983	985	987	989	991
	993	995	997	999	1001	1003	1005	1007	1009	1011	1013	1015	1017	1019
	1021	1023	1025	1027	1029	1031	1033	1035	1037	1039	1041	1043	1045	1047
	1049	1051	1053	1055	1057	1059	1061	1063	1065	1067	1069	1071	1073	1075
	1077	1079	1081	1083	1085	1087	1089	1091	1093	1095	1097	1099	1101	1103
	1105	1107	1109	1111	1113	1115	1117	1119	1121	1123	1125	1127	1129	1131
	1133	1135	1137	1139	1141	1143	1145	1147	1149	1151	1153	1155	1157	1159
	1161	1163	1165	1167	1169	1171	1173	1175	1177	1179	1181	1183	1185	1187
	1189	1191	1193	1195	1197	1199	1201	1203	1205	1207	1209	1211	1213	1215
	1217	1219	1221	1223	1225	1227	1229	1231	1233	1235	1237	1239	1241	1243
	1245	1247	1249	1251	1253	1255	1257	1259	1261	1263	1265	1267	1269	1271
	1273	1275	1277	1279	1281	1283	1285	1287	1289	1291	1293	1295	1297	1299
	1301	1303	1305	1307	1309	1311	1313	1315	1317	1319	1321	1323	1325	1327
	1329	1331	1333	1335	1337	1339	1341	1343	1345	1347	1349	1351	1353	1355
	1357	1359	1361	1363	1365	1367	1369	1371	1373	1375	1377	1379	1381	1383
	1385	1387	1389	1391	1393	1395	1397	1399	1401	1403	1405	1407	1409	1411
	1413	1415	1417	1419	1421	1423	1425	1427	1429	1431	1433	1435	1437	1439
	1441	1443	1445	1447	1449	1451	1453	1455	1457	1459	1461	1463	1465	1467
	1469	1471	1473	1475	1477	1479	1481	1483	1485	1487	1489	1491	1493	1495
	1497	1499	1501	1503	1505	1507	1509	1511	1513	1515	1517	1519	1521	1523
	1525	1527	1529	1531	1533	1535	1537	1539	1541	1543	1545	1547	1549	1551
	1553	1555	1557	1559	1561	1563	1565	1567	1569	1571	1573	1575	1577	1579
	1581	1583	1585	1587	1589	1591	1593	1595	1597	1599	1601	1603	1605	1607
	1609	1611	1613	1615	1617	1619	1621	1623	1625	1627	1629	1631	1633	1635
	1637	1639	1641	1643	1645	1647	1649	1651	1653	1655	1657	1659	1661	1663
	1665	1667	1669	1671	1673	1675	1677	1679	1681	1683	1685	1687	1689	1691
	1693	1695	1697	1699	1701	1703	1705	1707	1709	1711	1713	1715	1717	1719
	1721	1723	1725	1727	1729	1731	1733	1735	1737	1739	1741	1743	1745	1747
	1749	1751	1753	1755	1757	1759	1761	1763	1765	1767	1769	1771	1773	1775
	1777	1779	1781	1783	1785	1787	1789	1791	1793	1795	1797	1799	1801	1803
	1805	1807	1809	1811	1813	1815	1817	1819	1821	1823	1825	1827	1829	1831
	1833	1835	1837	1839	1841	1843	1845	1847	1849	1851	1853	1855	1857	1859
	1861	1863	1865	1867	1869	1871	1873	1875	1877	1879	1881	1883	1885	1887
	1889	1891	1893	1895	1897	1899	1901	1903	1905	1907	1909	1911	1913	1915
	1917	1919	1921	1923	1925	1927	1929	1931	1933	1935	1937	1939	1941	1943
	1945	1947	1949	1951	1953	1955	1957	1959	1961	1963	1965	1967	1969	1971
	1973	1975	1977	1979	1981	1983	1985	1987	1989	1991	1993	1995	1997	1999
	2001	2003	2005	2007	2009	2011	2013	2015	2017	2019	2021	2023	2025	2027
	2029	2031	2033	2035	2037	2039	2041	2043	2045	2047	2049	2051	2053	2055
	2057	2059	2061	2063	2065	2067	2069	2071	2073	2075	2077	2082	2087	
GO	84	86	96											
MFLD	116#	116	121#	121	123#	123	125#	125	127#	127	129#	129	131#	131

SEQ 1180

```
SEQ 1168   133#  133   135#  135   137#  137   139#  139   141#  141   143#  143   145#  145   SEQ 1181
           147#  147   149#  149   151#  151   153#  153   161#  161   163#  163   165#  165
           167#  167   169#  169   171#  171   173#  173   175#  175   177#  177   179#  179
           181#  181   183#  183   185#  185   187#  187   189#  189   191#  191   193#  193
           195#  195   197#  197   199#  199   201#  201   203#  203   205#  205   207#  207
           209#  209   211#  211   213#  213   215#  215   217#  217   219#  219   221#  221
           223#  223   225#  225   227#  227   229#  229   231#  231   233#  233   235#  235
           237#  237   239#  239   241#  241   243#  243   245#  245   247#  247   249#  249
           251#  251   253#  253   255#  255   257#  257   259#  259   261#  261   263#  263
           265#  265   267#  267   269#  269   271#  271   273#  273   275#  275   277#  277
           279#  279   281#  281   283#  283   285#  285   287#  287   289#  289   291#  291
           293#  293   295#  295   297#  297   299#  299   301#  301   303#  303   305#  305
           307#  307   309#  309   311#  311   313#  313   315#  315   317#  317   319#  319
           321#  321   323#  323   325#  325   327#  327   329#  329   331#  331   333#  333
           335#  335   337#  337   339#  339   341#  341   343#  343   345#  345   347#  347
           349#  349   351#  351   353#  353   355#  355   357#  357   359#  359   361#  361
           363#  363   365#  365   367#  367   369#  369   371#  371   373#  373   375#  375
           377#  377   379#  379   381#  381   383#  383   385#  385   387#  387   389#  389
           391#  391   393#  393   395#  395   397#  397   399#  399   401#  401   403#  403
           405#  405   407#  407   409#  409   411#  411   413#  413   415#  415   417#  417
           419#  419   421#  421   423#  423   425#  425   427#  427   429#  429   431#  431
           433#  433   435#  435   437#  437   439#  439   441#  441   443#  443   445#  445
           447#  447   449#  449   451#  451   453#  453   455#  455   457#  457   459#  459
           461#  461   463#  463   465#  465   467#  467   469#  469   471#  471   473#  473
           475#  475   477#  477   479#  479   481#  481   483#  483   485#  485   487#  487
           489#  489   491#  491   493#  493   495#  495   497#  497   499#  499   501#  501
           503#  503   505#  505   507#  507   509#  509   511#  511   513#  513   515#  515
           517#  517   519#  519   521#  521   523#  523   525#  525   527#  527   529#  529
           531#  531   533#  533   535#  535   537#  537   539#  539   541#  541   543#  543
           545#  545   547#  547   549#  549   551#  551   553#  553   555#  555   557#  557
           559#  559   561#  561   563#  563   565#  565   567#  567   569#  569   571#  571
           573#  573   575#  575   577#  577   579#  579   581#  581   583#  583   585#  585
           587#  587   589#  589   591#  591   593#  593   595#  595   597#  597   599#  599
           601#  601   603#  603   605#  605   607#  607   609#  609   611#  611   613#  613
           615#  615   617#  617   619#  619   621#  621   623#  623   625#  625   627#  627
           629#  629   631#  631   633#  633   635#  635   637#  637   639#  639   641#  641
           643#  643   645#  645   647#  647   649#  649   651#  651   653#  653   655#  655
           657#  657   659#  659   661#  661   663#  663   665#  665   667#  667   669#  669
           671#  671   673#  673   675#  675   677#  677   679#  679   681#  681   683#  683
           685#  685   687#  687   689#  689   691#  691   693#  693   695#  695   697#  697
           699#  699   701#  701   703#  703   705#  705   707#  707   709#  709   711#  711
           713#  713   715#  715   717#  717   719#  719   721#  721   723#  723   725#  725
           727#  727   729#  729   731#  731   733#  733   735#  735   737#  737   739#  739
           741#  741   743#  743   745#  745   747#  747   749#  749   751#  751   753#  753
           755#  755   757#  757   759#  759   761#  761   763#  763   765#  765   767#  767
           769#  769   771#  771   773#  773   775#  775   777#  777   779#  779   781#  781
           783#  783   785#  785   787#  787   789#  789   791#  791   793#  793   795#  795
           797#  797   799#  799   801#  801   803#  803   805#  805   807#  807   809#  809
           811#  811   813#  813   815#  815   817#  817   819#  819   821#  821   823#  823
           825#  825   827#  827   829#  829   831#  831   833#  833   835#  835   837#  837
           839#  839   841#  841   843#  843   845#  845   847#  847   849#  849   851#  851
           853#  853   855#  855   857#  857   859#  859   861#  861   863#  863   865#  865
           867#  867   869#  869   871#  871   873#  873   875#  875   877#  877   879#  879
```

```
SEQ 1169    881#  881    883#  883    885#  885    887#  887    889#  889    891#  891    893#  893    SEQ 1182
            895#  895    897#  897    899#  899    901#  901    903#  903    905#  905    907#  907
            909#  909    911#  911    913#  913    915#  915    917#  917    919#  919    921#  921
            923#  923    925#  925    927#  927    929#  929    931#  931    933#  933    935#  935
            937#  937    939#  939    941#  941    943#  943    945#  945    947#  947    949#  949
            951#  951    953#  953    955#  955    957#  957    959#  959    961#  961    963#  963
            965#  965    967#  967    969#  969    971#  971    973#  973    975#  975    977#  977
            979#  979    981#  981    983#  983    985#  985    987#  987    989#  989    991#  991
            993#  993    995#  995    997#  997    999#  999   1001# 1001   1003# 1003   1005# 1005
           1007# 1007   1009# 1009   1011# 1011   1013# 1013   1015# 1015   1017# 1017   1019# 1019
           1021# 1021   1023# 1023   1025# 1025   1027# 1027   1029# 1029   1031# 1031   1033# 1033
           1035# 1035   1037# 1037   1039# 1039   1041# 1041   1043# 1043   1045# 1045   1047# 1047
           1049# 1049   1051# 1051   1053# 1053   1055# 1055   1057# 1057   1059# 1059   1061# 1061
           1063# 1063   1065# 1065   1067# 1067   1069# 1069   1071# 1071   1073# 1073   1075# 1075
           1077# 1077   1079# 1079   1081# 1081   1083# 1083   1085# 1085   1087# 1087   1089# 1089
           1091# 1091   1093# 1093   1095# 1095   1097# 1097   1099# 1099   1101# 1101   1103# 1103
           1105# 1105   1107# 1107   1109# 1109   1111# 1111   1113# 1113   1115# 1115   1117# 1117
           1119# 1119   1121# 1121   1123# 1123   1125# 1125   1127# 1127   1129# 1129   1131# 1131
           1133# 1133   1135# 1135   1137# 1137   1139# 1139   1141# 1141   1143# 1143   1145# 1145
           1147# 1147   1149# 1149   1151# 1151   1153# 1153   1155# 1155   1157# 1157   1159# 1159
           1161# 1161   1163# 1163   1165# 1165   1167# 1167   1169# 1169   1171# 1171   1173# 1173
           1175# 1175   1177# 1177   1179# 1179   1181# 1181   1183# 1183   1185# 1185   1187# 1187
           1189# 1189   1191# 1191   1193# 1193   1195# 1195   1197# 1197   1199# 1199   1201# 1201
           1203# 1203   1205# 1205   1207# 1207   1209# 1209   1211# 1211   1213# 1213   1215# 1215
           1217# 1217   1219# 1219   1221# 1221   1223# 1223   1225# 1225   1227# 1227   1229# 1229
           1231# 1231   1233# 1233   1235# 1235   1237# 1237   1239# 1239   1241# 1241   1243# 1243
           1245# 1245   1247# 1247   1249# 1249   1251# 1251   1253# 1253   1255# 1255   1257# 1257
           1259# 1259   1261# 1261   1263# 1263   1265# 1265   1267# 1267   1269# 1269   1271# 1271
           1273# 1273   1275# 1275   1277# 1277   1279# 1279   1281# 1281   1283# 1283   1285# 1285
           1287# 1287   1289# 1289   1291# 1291   1293# 1293   1295# 1295   1297# 1297   1299# 1299
           1301# 1301   1303# 1303   1305# 1305   1307# 1307   1309# 1309   1311# 1311   1313# 1313
           1315# 1315   1317# 1317   1319# 1319   1321# 1321   1323# 1323   1325# 1325   1327# 1327
           1329# 1329   1331# 1331   1333# 1333   1335# 1335   1337# 1337   1339# 1339   1341# 1341
           1343# 1343   1345# 1345   1347# 1347   1349# 1349   1351# 1351   1353# 1353   1355# 1355
           1357# 1357   1359# 1359   1361# 1361   1363# 1363   1365# 1365   1367# 1367   1369# 1369
           1371# 1371   1373# 1373   1375# 1375   1377# 1377   1379# 1379   1381# 1381   1383# 1383
           1385# 1385   1387# 1387   1389# 1389   1391# 1391   1393# 1393   1395# 1395   1397# 1397
           1399# 1399   1401# 1401   1403# 1403   1405# 1405   1407# 1407   1409# 1409   1411# 1411
           1413# 1413   1415# 1415   1417# 1417   1419# 1419   1421# 1421   1423# 1423   1425# 1425
           1427# 1427   1429# 1429   1431# 1431   1433# 1433   1435# 1435   1437# 1437   1439# 1439
           1441# 1441   1443# 1443   1445# 1445   1447# 1447   1449# 1449   1451# 1451   1453# 1453
           1455# 1455   1457# 1457   1459# 1459   1461# 1461   1463# 1463   1465# 1465   1467# 1467
           1469# 1469   1471# 1471   1473# 1473   1475# 1475   1477# 1477   1479# 1479   1481# 1481
           1483# 1483   1485# 1485   1487# 1487   1489# 1489   1491# 1491   1493# 1493   1495# 1495
           1497# 1497   1499# 1499   1501# 1501   1503# 1503   1505# 1505   1507# 1507   1509# 1509
           1511# 1511   1513# 1513   1515# 1515   1517# 1517   1519# 1519   1521# 1521   1523# 1523
           1525# 1525   1527# 1527   1529# 1529   1531# 1531   1533# 1533   1535# 1535   1537# 1537
           1539# 1539   1541# 1541   1543# 1543   1545# 1545   1547# 1547   1549# 1549   1551# 1551
           1553# 1553   1555# 1555   1557# 1557   1559# 1559   1561# 1561   1563# 1563   1565# 1565
           1567# 1567   1569# 1569   1571# 1571   1573# 1573   1575# 1575   1577# 1577   1579# 1579
           1581# 1581   1583# 1583   1585# 1585   1587# 1587   1589# 1589   1591# 1591   1593# 1593
           1595# 1595   1597# 1597   1599# 1599   1601# 1601   1603# 1603   1605# 1605   1607# 1607
           1609# 1609   1611# 1611   1613# 1613   1615# 1615   1617# 1617   1619# 1619   1621# 1621
```

```
SEQ 1170  1623#  1623   1625#  1625   1627#  1627   1629#  1629   1631#  1631   1633#  1633   1635#  1635   SEQ 1183
          1637#  1637   1639#  1639   1641#  1641   1643#  1643   1645#  1645   1647#  1647   1649#  1649
          1651#  1651   1653#  1653   1655#  1655   1657#  1657   1659#  1659   1661#  1661   1663#  1663
          1665#  1665   1667#  1667   1669#  1669   1671#  1671   1673#  1673   1675#  1675   1677#  1677
          1679#  1679   1681#  1681   1683#  1683   1685#  1685   1687#  1687   1689#  1689   1691#  1691
          1693#  1693   1695#  1695   1697#  1697   1699#  1699   1701#  1701   1703#  1703   1705#  1705
          1707#  1707   1709#  1709   1711#  1711   1713#  1713   1715#  1715   1717#  1717   1719#  1719
          1721#  1721   1723#  1723   1725#  1725   1727#  1727   1729#  1729   1731#  1731   1733#  1733
          1735#  1735   1737#  1737   1739#  1739   1741#  1741   1743#  1743   1745#  1745   1747#  1747
          1749#  1749   1751#  1751   1753#  1753   1755#  1755   1757#  1757   1759#  1759   1761#  1761
          1763#  1763   1765#  1765   1767#  1767   1769#  1769   1771#  1771   1773#  1773   1775#  1775
          1777#  1777   1779#  1779   1781#  1781   1783#  1783   1785#  1785   1787#  1787   1789#  1789
          1791#  1791   1793#  1793   1795#  1795   1797#  1797   1799#  1799   1801#  1801   1803#  1803
          1805#  1805   1807#  1807   1809#  1809   1811#  1811   1813#  1813   1815#  1815   1817#  1817
          1819#  1819   1821#  1821   1823#  1823   1825#  1825   1827#  1827   1829#  1829   1831#  1831
          1833#  1833   1835#  1835   1837#  1837   1839#  1839   1841#  1841   1843#  1843   1845#  1845
          1847#  1847   1849#  1849   1851#  1851   1853#  1853   1855#  1855   1857#  1857   1859#  1859
          1861#  1861   1863#  1863   1865#  1865   1867#  1867   1869#  1869   1871#  1871   1873#  1873
          1875#  1875   1877#  1877   1879#  1879   1881#  1881   1883#  1883   1885#  1885   1887#  1887
          1889#  1889   1891#  1891   1893#  1893   1895#  1895   1897#  1897   1899#  1899   1901#  1901
          1903#  1903   1905#  1905   1907#  1907   1909#  1909   1911#  1911   1913#  1913   1915#  1915
          1917#  1917   1919#  1919   1921#  1921   1923#  1923   1925#  1925   1927#  1927   1929#  1929
          1931#  1931   1933#  1933   1935#  1935   1937#  1937   1939#  1939   1941#  1941   1943#  1943
          1945#  1945   1947#  1947   1949#  1949   1951#  1951   1953#  1953   1955#  1955   1957#  1957
          1959#  1959   1961#  1961   1963#  1963   1965#  1965   1967#  1967   1969#  1969   1971#  1971
          1973#  1973   1975#  1975   1977#  1977   1979#  1979   1981#  1981   1983#  1983   1985#  1985
          1987#  1987   1989#  1989   1991#  1991   1993#  1993   1995#  1995   1997#  1997   1999#  1999
          2001#  2001   2003#  2003   2005#  2005   2007#  2007   2009#  2009   2011#  2011   2013#  2013
          2015#  2015   2017#  2017   2019#  2019   2021#  2021   2023#  2023   2025#  2025   2027#  2027
          2029#  2029   2031#  2031   2033#  2033   2035#  2035   2037#  2037   2039#  2039   2041#  2041
          2043#  2043   2045#  2045   2047#  2047   2049#  2049   2051#  2051   2053#  2053   2055#  2055
          2057#  2057   2059#  2059   2061#  2061   2063#  2063   2065#  2065   2067#  2067   2069#  2069
          2071#  2071   2073#  2073   2075#  2075   2077#  2077   2082#  2082   2087#  2087
MWORD      116    121    123    125    127    129    131    133    135    137    139    141    143    145
           147    149    151    153    161    163    165    167    169    171    173    175    177    179
           181    183    185    187    189    191    193    195    197    199    201    203    205    207
           209    211    213    215    217    219    221    223    225    227    229    231    233    235
           237    239    241    243    245    247    249    251    253    255    257    259    261    263
           265    267    269    271    273    275    277    279    281    283    285    287    289    291
           293    295    297    299    301    303    305    307    309    311    313    315    317    319
           321    323    325    327    329    331    333    335    337    339    341    343    345    347
           349    351    353    355    357    359    361    363    365    367    369    371    373    375
           377    379    381    383    385    387    389    391    393    395    397    399    401    403
           405    407    409    411    413    415    417    419    421    423    425    427    429    431
           433    435    437    439    441    443    445    447    449    451    453    455    457    459
           461    463    465    467    469    471    473    475    477    479    481    483    485    487
           489    491    493    495    497    499    501    503    505    507    509    511    513    515
           517    519    521    523    525    527    529    531    533    535    537    539    541    543
           545    547    549    551    553    555    557    559    561    563    565    567    569    571
           573    575    577    579    581    583    585    587    589    591    593    595    597    599
           601    603    605    607    609    611    613    615    617    619    621    623    625    627
           629    631    633    635    637    639    641    643    645    647    649    651    653    655
           657    659    661    663    665    667    669    671    673    675    677    679    681    683
```

```
             685   687   689   691   693   695   697   699   701   703   705   707   709   711
             713   715   717   719   721   723   725   727   729   731   733   735   737   739
             741   743   745   747   749   751   753   755   757   759   761   763   765   767
             769   771   773   775   777   779   781   783   785   787   789   791   793   795
             797   799   801   803   805   807   809   811   813   815   817   819   821   823
             825   827   829   831   833   835   837   839   841   843   845   847   849   851
             853   855   857   859   861   863   865   867   869   871   873   875   877   879
             881   883   885   887   889   891   893   895   897   899   901   903   905   907
             909   911   913   915   917   919   921   923   925   927   929   931   933   935
             937   939   941   943   945   947   949   951   953   955   957   959   961   963
             965   967   969   971   973   975   977   979   981   983   985   987   989   991
             993   995   997   999  1001  1003  1005  1007  1009  1011  1013  1015  1017  1019
            1021  1023  1025  1027  1029  1031  1033  1035  1037  1039  1041  1043  1045  1047
            1049  1051  1053  1055  1057  1059  1061  1063  1065  1067  1069  1071  1073  1075
            1077  1079  1081  1083  1085  1087  1089  1091  1093  1095  1097  1099  1101  1103
            1105  1107  1109  1111  1113  1115  1117  1119  1121  1123  1125  1127  1129  1131
            1133  1135  1137  1139  1141  1143  1145  1147  1149  1151  1153  1155  1157  1159
            1161  1163  1165  1167  1169  1171  1173  1175  1177  1179  1181  1183  1185  1187
            1189  1191  1193  1195  1197  1199  1201  1203  1205  1207  1209  1211  1213  1215
            1217  1219  1221  1223  1225  1227  1229  1231  1233  1235  1237  1239  1241  1243
            1245  1247  1249  1251  1253  1255  1257  1259  1261  1263  1265  1267  1269  1271
            1273  1275  1277  1279  1281  1283  1285  1287  1289  1291  1293  1295  1297  1299
            1301  1303  1305  1307  1309  1311  1313  1315  1317  1319  1321  1323  1325  1327
            1329  1331  1333  1335  1337  1339  1341  1343  1345  1347  1349  1351  1353  1355
            1357  1359  1361  1363  1365  1367  1369  1371  1373  1375  1377  1379  1381  1383
            1385  1387  1389  1391  1393  1395  1397  1399  1401  1403  1405  1407  1409  1411
            1413  1415  1417  1419  1421  1423  1425  1427  1429  1431  1433  1435  1437  1439
            1441  1443  1445  1447  1449  1451  1453  1455  1457  1459  1461  1463  1465  1467
            1469  1471  1473  1475  1477  1479  1481  1483  1485  1487  1489  1491  1493  1495
            1497  1499  1501  1503  1505  1507  1509  1511  1513  1515  1517  1519  1521  1523
            1525  1527  1529  1531  1533  1535  1537  1539  1541  1543  1545  1547  1549  1551
            1553  1555  1557  1559  1561  1563  1565  1567  1569  1571  1573  1575  1577  1579
            1581  1583  1585  1587  1589  1591  1593  1595  1597  1599  1601  1603  1605  1607
            1609  1611  1613  1615  1617  1619  1621  1623  1625  1627  1629  1631  1633  1635
            1637  1639  1641  1643  1645  1647  1649  1651  1653  1655  1657  1659  1661  1663
            1665  1667  1669  1671  1673  1675  1677  1679  1681  1683  1685  1687  1689  1691
            1693  1695  1697  1699  1701  1703  1705  1707  1709  1711  1713  1715  1717  1719
            1721  1723  1725  1727  1729  1731  1733  1735  1737  1739  1741  1743  1745  1747
            1749  1751  1753  1755  1757  1759  1761  1763  1765  1767  1769  1771  1773  1775
            1777  1779  1781  1783  1785  1787  1789  1791  1793  1795  1797  1799  1801  1803
            1805  1807  1809  1811  1813  1815  1817  1819  1821  1823  1825  1827  1829  1831
            1833  1835  1837  1839  1841  1843  1845  1847  1849  1851  1853  1855  1857  1859
            1861  1863  1865  1867  1869  1871  1873  1875  1877  1879  1881  1883  1885  1887
            1889  1891  1893  1895  1897  1899  1901  1903  1905  1907  1909  1911  1913  1915
            1917  1919  1921  1923  1925  1927  1929  1931  1933  1935  1937  1939  1941  1943
            1945  1947  1949  1951  1953  1955  1957  1959  1961  1963  1965  1967  1969  1971
            1973  1975  1977  1979  1981  1983  1985  1987  1989  1991  1993  1995  1997  1999
            2001  2003  2005  2007  2009  2011  2013  2015  2017  2019  2021  2023  2025  2027
            2029  2031  2033  2035  2037  2039  2041  2043  2045  2047  2049  2051  2053  2055
            2057  2059  2061  2063  2065  2067  2069  2071  2073  2075  2077  2082  2087
RTN           87   104
SCOPER        98
```

```
                                        TITLE   DFPTA6
                                        SUBTTL  EBUS/MPROC 2901 Tests (Part 3)

                                        SALL

                                        SEARCH  DFPTA,MONSYM

                                ; INTERN's

                                        ENTRY   TSTA32

                                ; EXTERN's

                                        EXTERN  TSTA36,TSTA37,TSTA40,TSTA41

                                ; EXTERN's located in DFPTA1.MAC (Mainline code..)

                                        EXTERN  TLOAD,TRACE,TSTSUB,AEXEC,AAPNT


                                ;#*********************************************************************
                                ; Z6 - Address for use in DDT
                                ;#*********************************************************************

 000000'                        Z6:                                     ; address of 00000'

```

```
                                                 SUBTTL  Register Interference Tests

                                         ;#***********************************************************************
                                         ;* TEST 32 - 2901 RAM Register Interference Test
                                         ;
                                         ; Description:  Load Register 0 with test data.  Then write  data
                                         ;               to Q-Register and Registers 1-15 with all combin-
                                         ;               ations of source operand, ALU function, and dest-
                                         ;               ination field (other than RAM function  specified
                                         ;               as loading Register 0) and  verify  that contents
                                         ;               of Register 0 did not change.
                                         ;
                                         ;               Repeat with Register 1-15.
                                         ;
                                         ; Procedure:    Clear Port
                                         ;               Load microcode/set start address 1
                                         ;               Start microcode (will halt with a parity error at
                                         ;                 completion of test).
                                         ;
                                         ;               For each register 0-15, call  subroutine  to  put
                                         ;               nonzero data in each RAM  and  Q registers.  Then
                                         ;               zero the register and execute a  block  of  micro
                                         ;               instructions.  If the register goes non-zero, the
                                         ;               microsequencer goes to the  error  routine.  When
                                         ;               done, read the LAR and verify that  the microcode
                                         ;               stopped in the right place.
                                         ;
                                         ;               Address (error occurred):      4100
                                         ;               Address (done successfully):  5000
                                         ;
                                         ; Failure:      ---
                                         ;#***********************************************************************

                                         ; Test data

000000' 254 00 0 00 000011'          TSTA32: JRST    TG32                    ; go start test
000001' 420402  000032                      EBUS!ALU!NDMP!ZALU!32           ; test mask
000002' 000032' 004351'                     T32M,,[ASCIZ ^2901 RAM Register Interference Test^]
000003' 004361' 000000                      [EXP MLAST!E23],,0
000004' 000000000000#                       TSTA36                          ; failure test table
000005' 000000000000#                       TSTA37                          ;   ...
000006' 000000000000#                       TSTA40
000007' 000000000000#                       TSTA41
000010' 777777  777777                      -1

                                         ; Start test

000011' 201 00 0 00 000000'          TG32:   MOVEI   Z6                      ; get address of module start
000012' 260 17 0 00 000000*                 GO      TRACE                   ; handle trace output
000013' 201 01 0 00 000032'                 MOVEI   1,T32M                  ; set up microcode address
000014' 260 17 0 00 000000*                 GO      TLOAD                   ; load/verify it
000015' 263 17 0 00 000000                  RTN                             ; failed - exit test
```

```

                                        ; 1st segment of test (Segment A) - Start up port and wait till it stops

 000016' 400 15 0 00 000000             TA32:   SETZ    ERFLG,                  ; clear error flag
 000017' 201 01 0 00 000040                     MOVEI   1,^D32                  ; maximum delay before timeout (msec)
 000020' 201 02 0 00 004100                     MOVEI   2,4100                  ; error address
 000021' 201 03 0 00 005000                     MOVEI   3,5000                  ; correct address
 000022' 201 04 0 00 000001                     MOVEI   4,1                     ; starting address
 000023' 260 17 0 00 000000*                    GO      AEXEC                   ; execute ALU type test
 000024' 474 15 0 00 000000                     SETO    ERFLG,                  ; error occurred
 000025' 027 00 0 00 000031'                    SCOPER  MA32                    ; print error message
 000026' 254 00 0 00 000016'                    JRST    TA32                    ; loop on error
 000027' 254 00 0 00 000030'                    JRST    TX32                    ; altmode exit

                                        ; End of test

 000030' 263 17 0 00 000000             TX32:   RTN                             ; return

                                        ; Error message - Microcode did not stop in the right place

 000031' 000000000000#                  MA32:   LAST!CALL!TXALL!AAPNT           ; print results of ALU type test
```

```


                                    ; Microcode:

000032' 000010  000000              T32M:   MWORD   <ADDR=0,JMAP,J=1000>                    ; 0
000033' 000000  000040
000034' 000100  000000                      MWORD   <JZ>                                    ; 1
000035' 000000  000000

                                    ;       Subroutine  to  initialize  RAM  and  Q  register
                                    ;       contents

000036' 000200  000022                      MWORD   <CONT,B=0,SDO,OR,D=0,SKCN,MGC=22>       ; 2
000037' 730000  240340
000040' 000300  000023                      MWORD   <CONT,B=0,SDO,OR,D=2,SKCN,MGC=23>       ; 3
000041' 732000  240340
000042' 000400  000024                      MWORD   <CONT,B=1,SDO,OR,D=2,SKCN,MGC=24>       ; 4
000043' 732000  640340
000044' 000500  000025                      MWORD   <CONT,B=2,SDO,OR,D=2,SKCN,MGC=25>       ; 5
000045' 732001  240340
000046' 000600  000026                      MWORD   <CONT,B=3,SDO,OR,D=2,SKCN,MGC=26>       ; 6
000047' 732001  640340
000050' 000700  000027                      MWORD   <CONT,B=4,SDO,OR,D=2,SKCN,MGC=27>       ; 7
000051' 732002  240340
000052' 001000  000030                      MWORD   <CONT,B=5,SDO,OR,D=2,SKCN,MGC=30>       ; 10
000053' 732002  640340
000054' 001100  000031                      MWORD   <CONT,B=6,SDO,OR,D=2,SKCN,MGC=31>       ; 11
000055' 732003  240340
000056' 001200  000032                      MWORD   <CONT,B=7,SDO,OR,D=2,SKCN,MGC=32>       ; 12
000057' 732003  640340
000060' 001300  000033                      MWORD   <CONT,B=10,SDO,OR,D=2,SKCN,MGC=33>      ; 13
000061' 732004  240340
000062' 001400  000034                      MWORD   <CONT,B=11,SDO,OR,D=2,SKCN,MGC=34>      ; 14
000063' 732004  640340
000064' 001500  000035                      MWORD   <CONT,B=12,SDO,OR,D=2,SKCN,MGC=35>      ; 15
000065' 732005  240340
000066' 001600  000036                      MWORD   <CONT,B=13,SDO,OR,D=2,SKCN,MGC=36>      ; 16
000067' 732005  640340
000070' 001700  000037                      MWORD   <CONT,B=14,SDO,OR,D=2,SKCN,MGC=37>      ; 17
000071' 732006  240340
000072' 002000  000040                      MWORD   <CONT,B=15,SDO,OR,D=2,SKCN,MGC=40>      ; 20
000073' 732006  640340
000074' 002100  000041                      MWORD   <CONT,B=16,SDO,OR,D=2,SKCN,MGC=41>      ; 21
000075' 732007  240340
000076' 002200  000042                      MWORD   <CONT,B=17,SDO,OR,D=2,SKCN,MGC=42>      ; 22
000077' 732007  640340
000100' 002300  000000                      MWORD   <CRTN,D=1>                              ; 23
000101' 001000  000240

                                    ;       Error checking routine - Register 0

000102' 400040  020000                      MWORD   <ADDR=4000,CJV,J=4002,SOA,OR,D=1,CENA,CCFZ> ; 4000
000103' 431400  020140
000104' 400140  010000                      MWORD   <JMAP,J=4001,BAD>                           ; 4001 error
000105' 000000  000041
000106' 400200  000000                      MWORD   <CRTN>                                      ; 4002 return
```

```
000107' 000000  000240

                                ;       Error checking routine - Register 1

000110' 400340  050000                  MWORD   <CJV,J=4005,A=1,SOA,OR,D=1,CENA,CCFZ>   ; 4003
000111' 431410  020140
000112' 400440  040000                  MWORD   <JMAP,J=4004,BAD>                       ; 4004 error
000113' 000000  000041
000114' 400500  000000                  MWORD   <CRTN>                                  ; 4005 return
000115' 000000  000240

                                ;       Error checking routine - Register 2

000116' 400640  100000                  MWORD   <CJV,J=4010,A=2,SOA,OR,D=1,CENA,CCFZ>   ; 4006
000117' 431420  020140
000120' 400740  070000                  MWORD   <JMAP,J=4007,BAD>                       ; 4007 error
000121' 000000  000041
000122' 401000  000000                  MWORD   <CRTN>                                  ; 4010 return
000123' 000000  000240

                                ;       Error checking routine - Register 3

000124' 401140  130000                  MWORD   <CJV,J=4013,A=3,SOA,OR,D=1,CENA,CCFZ>   ; 4011
000125' 431430  020140
000126' 401240  120000                  MWORD   <JMAP,J=4012,BAD>                       ; 4012 error
000127' 000000  000041
000130' 401300  000000                  MWORD   <CRTN>                                  ; 4013 return
000131' 000000  000240

                                ;       Error checking routine - Register 4

000132' 401440  160000                  MWORD   <CJV,J=4016,A=4,SOA,OR,D=1,CENA,CCFZ>   ; 4014
000133' 431440  020140
000134' 401540  150000                  MWORD   <JMAP,J=4015,BAD>                       ; 4015 error
000135' 000000  000041
000136' 401600  000000                  MWORD   <CRTN>                                  ; 4016 return
000137' 000000  000240

                                ;       Error checking routine - Register 5

000140' 401740  210000                  MWORD   <CJV,J=4021,A=5,SOA,OR,D=1,CENA,CCFZ>   ; 4017
000141' 431450  020140
000142' 402040  200000                  MWORD   <JMAP,J=4020,BAD>                       ; 4020 error
000143' 000000  000041
000144' 402100  000000                  MWORD   <CRTN>                                  ; 4021 return
000145' 000000  000240

                                ;       Error checking routine - Register 6

000146' 402240  240000                  MWORD   <CJV,J=4024,A=6,SOA,OR,D=1,CENA,CCFZ>   ; 4022
000147' 431460  020140
000150' 402340  230000                  MWORD   <JMAP,J=4023,BAD>                       ; 4023 error
000151' 000000  000041
000152' 402400  000000                  MWORD   <CRTN>                                  ; 4024 return
000153' 000000  000240
```

```

                                         ;       Error checking routine - Register 7

000154' 402540  270000                           MWORD   <CJV,J=4027,A=7,SOA,OR,D=1,CENA,CCFZ>   ; 4025
000155' 431470  020140                           MWORD   <JMAP,J=4026,BAD>                       ; 4026 error
000156' 402640  260000                           MWORD   <CRTN>                                  ; 4027 return
000157' 000000  000041
000160' 402700  000000
000161' 000000  000240

                                         ;       Error checking routine - Register 10

000162' 403040  320000                           MWORD   <CJV,J=4032,A=10,SOA,OR,D=1,CENA,CCFZ>  ; 4030
000163' 431500  020140                           MWORD   <JMAP,J=4031,BAD>                       ; 4031 error
000164' 403140  310000                           MWORD   <CRTN>                                  ; 4032 return
000165' 000000  000041
000166' 403200  000000
000167' 000000  000240

                                         ;       Error checking routine - Register 11

000170' 403340  350000                           MWORD   <CJV,J=4035,A=11,SOA,OR,D=1,CENA,CCFZ>  ; 4033
000171' 431510  020140                           MWORD   <JMAP,J=4034,BAD>                       ; 4034 error
000172' 403440  340000                           MWORD   <CRTN>                                  ; 4035 return
000173' 000000  000041
000174' 403500  000000
000175' 000000  000240

                                         ;       Error checking routine - Register 12

000176' 403640  400000                           MWORD   <CJV,J=4040,A=12,SOA,OR,D=1,CENA,CCFZ>  ; 4036
000177' 431520  020140                           MWORD   <JMAP,J=4037,BAD>                       ; 4037 error
000200' 403740  370000                           MWORD   <CRTN>                                  ; 4040 return
000201' 000000  000041
000202' 404000  000000
000203' 000000  000240

                                         ;       Error checking routine - Register 13

000204' 404140  430000                           MWORD   <CJV,J=4043,A=13,SOA,OR,D=1,CENA,CCFZ>  ; 4041
000205' 431530  020140                           MWORD   <JMAP,J=4042,BAD>                       ; 4042 error
000206' 404240  420000                           MWORD   <CRTN>                                  ; 4043 return
000207' 000000  000041
000210' 404300  000000
000211' 000000  000240

                                         ;       Error checking routine - Register 14

000212' 404440  460000                           MWORD   <CJV,J=4046,A=14,SOA,OR,D=1,CENA,CCFZ>  ; 4044
000213' 431540  020140                           MWORD   <JMAP,J=4045,BAD>                       ; 4045 error
000214' 404540  450000                           MWORD   <CRTN>                                  ; 4046 return
000215' 000000  000041
000216' 404600  000000
000217' 000000  000240

```

```
                                  ;        Error checking routine - Register 15

000220' 404740  510000                     MWORD   <CJV,J=4051,A=15,SOA,OR,D=1,CENA,CCFZ>  ; 4047
000221' 431550  020140
000222' 405040  500000                     MWORD   <JMAP,J=4050,BAD>                       ; 4050 error
000223' 000000  000041
000224' 405100  000000                     MWORD   <CRTN>                                  ; 4051 return
000225' 000000  000240

                                  ;        Error checking routine - Register 16

000226' 405240  540000                     MWORD   <CJV,J=4054,A=16,SOA,OR,D=1,CENA,CCFZ>  ; 4052
000227' 431560  020140
000230' 405340  530000                     MWORD   <JMAP,J=4053,BAD>                       ; 4053 error
000231' 000000  000041
000232' 405400  000000                     MWORD   <CRTN>                                  ; 4054 return
000233' 000000  000240

                                  ;        Error checking routine - Register 17

000234' 405540  570000                     MWORD   <CJV,J=4057,A=17,SOA,OR,D=1,CENA,CCFZ>  ; 4055
000235' 431570  020140
000236' 405640  560000                     MWORD   <JMAP,J=4056,BAD>                       ; 4056 error
000237' 000000  000041
000240' 405700  000000                     MWORD   <CRTN>                                  ; 4057 return
000241' 000000  000240

                                  ;        Error/Done locations

000242' 410041  000000                     MWORD   <ADDR=4100,JMAP,J=4100,BAD>             ; 4100 error
000243' 000000  000041
000244' 500050  000000                     MWORD   <ADDR=5000,JMAP,J=5000,BAD>             ; 5000 done ok
000245' 000000  000041

                                  ;        Register 0

000246' 100000  020000                     MWORD   <ADDR=1000,CJS,J=2>                         ; 1000
000247' 000000  000020
000250' 100100  000000                     MWORD   <CONT,A=0,B=0,SOA,AND,D=2>                  ; 1001
000251' 442000  000340

000252' 100200  001777                     MWORD   <CONT,A=0,B=10,SAQ,PLUS,D=0,SKCN,MGC=1777>  ; 1002
000253' 000004  240340
000254' 100340  000000                     MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                 ; 1003
000255' 431000  000020
000256' 100400  001777                     MWORD   <CONT,A=1,B=11,SAB,SMIN,D=1,SKCN,MGC=1777>  ; 1004
000257' 111014  640340
000260' 100540  000000                     MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                 ; 1005
000261' 431000  000020
000262' 100600  001777                     MWORD   <CONT,A=2,B=12,SAQ,RMIN,D=2,SKCN,MGC=1777>  ; 1006
000263' 022025  240340
000264' 100740  000000                     MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                 ; 1007
000265' 431000  000020
000266' 101000  001777                     MWORD   <CONT,A=3,B=13,SAB,OR,D=3,SKCN,MGC=1777>    ; 1010
000267' 133035  640340
```

```
000270' 101140  000000          MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1011
000271' 431000  000020
000272' 101200  001777          MWORD   <CONT,A=4,B=14,SAQ,AND,D=4,SKCN,MGC=1777>       ; 1012
000273' 044046  240340
000274' 101340  000000          MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1013
000275' 431000  000020
000276' 101400  001777          MWORD   <CONT,A=5,B=15,SAB,NAND,D=5,SKCN,MGC=1777>      ; 1014
000277' 155056  640340
000300' 101540  000000          MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1015
000301' 431000  000020
000302' 101600  001777          MWORD   <CONT,A=6,B=16,SAQ,XOR,D=6,SKCN,MGC=1777>       ; 1016
000303' 066067  240340
000304' 101740  000000          MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1017
000305' 431000  000020
000306' 102000  001777          MWORD   <CONT,A=7,B=17,SAB,XNOR,D=7,SKCN,MGC=1777>      ; 1020
000307' 177077  640340
000310' 102140  000000          MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1021
000311' 431000  000020

000312' 102200  001777          MWORD   <CONT,A=0,B=10,SOQ,XNOR,D=0,SKCN,MGC=1777>      ; 1022
000313' 270004  240340
000314' 102340  000000          MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1023
000315' 431000  000020
000316' 102400  001777          MWORD   <CONT,A=1,B=11,SOB,XOR,D=1,SKCN,MGC=1777>       ; 1024
000317' 361014  640340
000320' 102540  000000          MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1025
000321' 431000  000020
000322' 102600  001777          MWORD   <CONT,A=2,B=12,SOQ,NAND,D=2,SKCN,MGC=1777>      ; 1026
000323' 252025  240340
000324' 102740  000000          MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1027
000325' 431000  000020
000326' 103000  001777          MWORD   <CONT,A=3,B=13,SOB,AND,D=3,SKCN,MGC=1777>       ; 1030
000327' 343035  640340
000330' 103140  000000          MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1031
000331' 431000  000020
000332' 103200  001777          MWORD   <CONT,A=4,B=14,SOQ,OR,D=4,SKCN,MGC=1777>        ; 1032
000333' 234046  240340
000334' 103340  000000          MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1033
000335' 431000  000020
000336' 103400  001777          MWORD   <CONT,A=5,B=15,SOB,RMIN,D=5,SKCN,MGC=1777>      ; 1034
000337' 325056  640340
000340' 103540  000000          MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1035
000341' 431000  000020
000342' 103600  001777          MWORD   <CONT,A=6,B=16,SOQ,SMIN,D=6,SKCN,MGC=1777>      ; 1036
000343' 216067  240340
000344' 103740  000000          MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1037
000345' 431000  000020
000346' 104000  001777          MWORD   <CONT,A=7,B=17,SOB,PLUS,D=7,SKCN,MGC=1777>      ; 1040
000347' 307077  640340
000350' 104140  000000          MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1041
000351' 431000  000020

000352' 104200  001777          MWORD   <CONT,A=0,B=10,SOA,PLUS,D=0,SKCN,MGC=1777>      ; 1042
000353' 400004  240340
000354' 104340  000000          MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1043
```

```
000355' 431000  000020
000356' 104400  001777                  MWORD   <CONT,A=1,B=11,SDA,SMIN,D=1,SKCN,MGC=1777>  ; 1044
000357' 511014  640340
000360' 104540  000000                  MWORD   <CJS,J=4000,A=0,S0A,OR,D=1>                 ; 1045
000361' 431000  000020
000362' 104600  001777                  MWORD   <CONT,A=2,B=12,S0A,RMIN,D=2,SKCN,MGC=1777>  ; 1046
000363' 422025  240340
000364' 104740  000000                  MWORD   <CJS,J=4000,A=0,S0A,OR,D=1>                 ; 1047
000365' 431000  000020
000366' 105000  001777                  MWORD   <CONT,A=3,B=13,SDA,OR,D=3,SKCN,MGC=1777>    ; 1050
000367' 533035  640340
000370' 105140  000000                  MWORD   <CJS,J=4000,A=0,S0A,OR,D=1>                 ; 1051
000371' 431000  000020
000372' 105200  001777                  MWORD   <CONT,A=4,B=14,S0A,AND,D=4,SKCN,MGC=1777>   ; 1052
000373' 444046  240340
000374' 105340  000000                  MWORD   <CJS,J=4000,A=0,S0A,OR,D=1>                 ; 1053
000375' 431000  000020
000376' 105400  001777                  MWORD   <CONT,A=5,B=15,SDA,NAND,D=5,SKCN,MGC=1777>  ; 1054
000377' 555056  640340
000400' 105540  000000                  MWORD   <CJS,J=4000,A=0,S0A,OR,D=1>                 ; 1055
000401' 431000  000020
000402' 105600  001777                  MWORD   <CONT,A=6,B=16,S0A,XOR,D=6,SKCN,MGC=1777>   ; 1056
000403' 466067  240340
000404' 105740  000000                  MWORD   <CJS,J=4000,A=0,S0A,OR,D=1>                 ; 1057
000405' 431000  000020
000406' 106000  001777                  MWORD   <CONT,A=7,B=17,SDA,XNOR,D=7,SKCN,MGC=1777>  ; 1060
000407' 577077  640340
000410' 106140  000000                  MWORD   <CJS,J=4000,A=0,S0A,OR,D=1>                 ; 1061
000411' 431000  000020

000412' 106200  001777                  MWORD   <CONT,A=0,B=10,SDQ,XNOR,D=0,SKCN,MGC=1777>  ; 1062
000413' 670004  240340
000414' 106340  000000                  MWORD   <CJS,J=4000,A=0,S0A,OR,D=1>                 ; 1063
000415' 431000  000020
000416' 106400  001777                  MWORD   <CONT,A=1,B=11,SD0,XOR,D=1,SKCN,MGC=1777>   ; 1064
000417' 761014  640340
000420' 106540  000000                  MWORD   <CJS,J=4000,A=0,S0A,OR,D=1>                 ; 1065
000421' 431000  000020
000422' 106600  001777                  MWORD   <CONT,A=2,B=12,SDQ,NAND,D=2,SKCN,MGC=1777>  ; 1066
000423' 652025  240340
000424' 106740  000000                  MWORD   <CJS,J=4000,A=0,S0A,OR,D=1>                 ; 1067
000425' 431000  000020
000426' 107000  001777                  MWORD   <CONT,A=3,B=13,SD0,AND,D=3,SKCN,MGC=1777>   ; 1070
000427' 743035  640340
000430' 107140  000000                  MWORD   <CJS,J=4000,A=0,S0A,OR,D=1>                 ; 1071
000431' 431000  000020
000432' 107200  001777                  MWORD   <CONT,A=4,B=14,SDQ,OR,D=4,SKCN,MGC=1777>    ; 1072
000433' 634046  240340
000434' 107340  000000                  MWORD   <CJS,J=4000,A=0,S0A,OR,D=1>                 ; 1073
000435' 431000  000020
000436' 107400  001777                  MWORD   <CONT,A=5,B=15,SD0,RMIN,D=5,SKCN,MGC=1777>  ; 1074
000437' 725056  640340
000440' 107540  000000                  MWORD   <CJS,J=4000,A=0,S0A,OR,D=1>                 ; 1075
000441' 431000  000020
000442' 107600  001777                  MWORD   <CONT,A=6,B=16,SDQ,SMIN,D=6,SKCN,MGC=1777>  ; 1076
```

```
000443' 616067  240340
000444' 107740  000000                  MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1077
000445' 431000  000020
000446' 110000  001777                  MWORD   <CONT,A=7,B=17,SDO,PLUS,D=7,SKCN,MGC=1777>      ; 1100
000447' 707077  640340
000450' 110140  000000                  MWORD   <CJS,J=4000,A=0,SOA,OR,D=1>                     ; 1101
000451' 431000  000020

                                ;       Now do register 1

000452' 110200  020000                  MWORD   <CJS,J=2>                                       ; 1102
000453' 000000  000020
000454' 110300  000000                  MWORD   <CONT,A=0,B=1,SOA,AND,D=2>                      ; 1103
000455' 442000  400340

000456' 110400  001777                  MWORD   <CONT,A=1,B=10,SAQ,PLUS,D=0,SKCN,MGC=1777>      ; 1104
000457' 000014  240340
000460' 110540  030000                  MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                     ; 1105
000461' 431010  000020
000462' 110600  001777                  MWORD   <CONT,A=1,B=11,SAB,SMIN,D=1,SKCN,MGC=1777>      ; 1106
000463' 111014  640340
000464' 110740  030000                  MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                     ; 1107
000465' 431010  000020
000466' 111000  001777                  MWORD   <CONT,A=2,B=12,SAQ,RMIN,D=2,SKCN,MGC=1777>      ; 1110
000467' 022025  240340
000470' 111140  030000                  MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                     ; 1111
000471' 431010  000020
000472' 111200  001777                  MWORD   <CONT,A=3,B=13,SAB,OR,D=3,SKCN,MGC=1777>        ; 1112
000473' 133035  640340
000474' 111340  030000                  MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                     ; 1113
000475' 431010  000020
000476' 111400  001777                  MWORD   <CONT,A=4,B=14,SAQ,AND,D=4,SKCN,MGC=1777>       ; 1114
000477' 044046  240340
000500' 111540  030000                  MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                     ; 1115
000501' 431010  000020
000502' 111600  001777                  MWORD   <CONT,A=5,B=15,SAB,NAND,D=5,SKCN,MGC=1777>      ; 1116
000503' 155056  640340
000504' 111740  030000                  MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                     ; 1117
000505' 431010  000020
000506' 112000  001777                  MWORD   <CONT,A=6,B=16,SAQ,XOR,D=6,SKCN,MGC=1777>       ; 1120
000507' 066067  240340
000510' 112140  030000                  MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                     ; 1121
000511' 431010  000020
000512' 112200  001777                  MWORD   <CONT,A=7,B=17,SAB,XNOR,D=7,SKCN,MGC=1777>      ; 1122
000513' 177077  640340
000514' 112340  030000                  MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                     ; 1123
000515' 431010  000020

000516' 112400  001777                  MWORD   <CONT,A=1,B=10,SOQ,XNOR,D=0,SKCN,MGC=1777>      ; 1124
000517' 270014  240340
000520' 112540  030000                  MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                     ; 1125
000521' 431010  000020
000522' 112600  001777                  MWORD   <CONT,A=1,B=11,SOB,XOR,D=1,SKCN,MGC=1777>       ; 1126
000523' 361014  640340
000524' 112740  030000                  MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                     ; 1127
```

```
485  000525' 431010  000020
486  000526' 113000  001777          MWORD   <CONT,A=2,B=12,SOQ,NAND,D=2,SKCN,MGC=1777>  ; 1130
487  000527' 252025  240340
488  000530' 113140  030000          MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                 ; 1131
489  000531' 431010  000020
490  000532' 113200  001777          MWORD   <CONT,A=3,B=13,SOB,AND,D=3,SKCN,MGC=1777>   ; 1132
491  000533' 343035  640340
492  000534' 113340  030000          MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                 ; 1133
493  000535' 431010  000020
494  000536' 113400  001777          MWORD   <CONT,A=4,B=14,SOQ,OR,D=4,SKCN,MGC=1777>    ; 1134
495  000537' 234046  240340
496  000540' 113540  030000          MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                 ; 1135
497  000541' 431010  000020
498  000542' 113600  001777          MWORD   <CONT,A=5,B=15,SOB,RMIN,D=5,SKCN,MGC=1777>  ; 1136
499  000543' 325056  640340
500  000544' 113740  030000          MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                 ; 1137
501  000545' 431010  000020
502  000546' 114000  001777          MWORD   <CONT,A=6,B=16,SOQ,SMIN,D=6,SKCN,MGC=1777>  ; 1140
503  000547' 216067  240340
504  000550' 114140  030000          MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                 ; 1141
505  000551' 431010  000020
506  000552' 114200  001777          MWORD   <CONT,A=7,B=17,SOB,PLUS,D=7,SKCN,MGC=1777>  ; 1142
507  000553' 307077  640340
508  000554' 114340  030000          MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                 ; 1143
509  000555' 431010  000020
510
511  000556' 114400  001777          MWORD   <CONT,A=1,B=10,SOA,PLUS,D=0,SKCN,MGC=1777>  ; 1144
512  000557' 400014  240340
513  000560' 114540  030000          MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                 ; 1145
514  000561' 431010  000020
515  000562' 114600  001777          MWORD   <CONT,A=1,B=11,SDA,SMIN,D=1,SKCN,MGC=1777>  ; 1146
516  000563' 511014  640340
517  000564' 114740  030000          MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                 ; 1147
518  000565' 431010  000020
519  000566' 115000  001777          MWORD   <CONT,A=2,B=12,SOA,RMIN,D=2,SKCN,MGC=1777>  ; 1150
520  000567' 422025  240340
521  000570' 115140  030000          MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                 ; 1151
522  000571' 431010  000020
523  000572' 115200  001777          MWORD   <CONT,A=3,B=13,SDA,OR,D=3,SKCN,MGC=1777>    ; 1152
524  000573' 533035  640340
525  000574' 115340  030000          MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                 ; 1153
526  000575' 431010  000020
527  000576' 115400  001777          MWORD   <CONT,A=4,B=14,SOA,AND,D=4,SKCN,MGC=1777>   ; 1154
528  000577' 444046  240340
529  000600' 115540  030000          MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                 ; 1155
530  000601' 431010  000020
531  000602' 115600  001777          MWORD   <CONT,A=5,B=15,SDA,NAND,D=5,SKCN,MGC=1777>  ; 1156
532  000603' 555056  640340
533  000604' 115740  030000          MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                 ; 1157
534  000605' 431010  000020
535  000606' 116000  001777          MWORD   <CONT,A=6,B=16,SOA,XOR,D=6,SKCN,MGC=1777>   ; 1160
536  000607' 466067  240340
537  000610' 116140  030000          MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                 ; 1161
538  000611' 431010  000020
539  000612' 116200  001777          MWORD   <CONT,A=7,B=17,SDA,XNOR,D=7,SKCN,MGC=1777>  ; 1162
```

```
000613' 577077  640340
000614' 116340  030000                    MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                        ; 1163
000615' 431010  000020

000616' 116400  001777                    MWORD   <CONT,A=1,B=10,SDQ,XNOR,D=0,SKCN,MGC=1777>  ; 1164
000617' 670014  240340
000620' 116540  030000                    MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                        ; 1165
000621' 431010  000020
000622' 116600  001777                    MWORD   <CONT,A=1,B=11,SDO,XOR,D=1,SKCN,MGC=1777>   ; 1166
000623' 761014  640340
000624' 116740  030000                    MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                        ; 1167
000625' 431010  000020
000626' 117000  001777                    MWORD   <CONT,A=2,B=12,SDQ,NAND,D=2,SKCN,MGC=1777>  ; 1170
000627' 652025  240340
000630' 117140  030000                    MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                        ; 1171
000631' 431010  000020
000632' 117200  001777                    MWORD   <CONT,A=3,B=13,SDO,AND,D=3,SKCN,MGC=1777>   ; 1172
000633' 743035  640340
000634' 117340  030000                    MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                        ; 1173
000635' 431010  000020
000636' 117400  001777                    MWORD   <CONT,A=4,B=14,SDQ,OR,D=4,SKCN,MGC=1777>    ; 1174
000637' 634046  240340
000640' 117540  030000                    MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                        ; 1175
000641' 431010  000020
000642' 117600  001777                    MWORD   <CONT,A=5,B=15,SDO,RMIN,D=5,SKCN,MGC=1777>  ; 1176
000643' 725056  640340
000644' 117740  030000                    MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                        ; 1177
000645' 431010  000020
000646' 120000  001777                    MWORD   <CONT,A=6,B=16,SDQ,SMIN,D=6,SKCN,MGC=1777>  ; 1200
000647' 616067  240340
000650' 120140  030000                    MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                        ; 1201
000651' 431010  000020
000652' 120200  001777                    MWORD   <CONT,A=7,B=17,SDO,PLUS,D=7,SKCN,MGC=1777>  ; 1202
000653' 707077  640340
000654' 120340  030000                    MWORD   <CJS,J=4003,A=1,SOA,OR,D=1>                        ; 1203
000655' 431010  000020


                                  ;       Now do register 2

000656' 120400  020000                    MWORD   <CJS,J=2>                                          ; 1204
000657' 000000  000020
000660' 120500  000000                    MWORD   <CONT,A=0,B=2,SOA,AND,D=2>                         ; 1205
000661' 442001  000340

000662' 120600  001777                    MWORD   <CONT,A=2,B=10,SAQ,PLUS,D=0,SKCN,MGC=1777>  ; 1206
000663' 000024  240340
000664' 120740  060000                    MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                        ; 1207
000665' 431020  000020
000666' 121000  001777                    MWORD   <CONT,A=1,B=11,SAB,SMIN,D=1,SKCN,MGC=1777>  ; 1210
000667' 111014  640340
000670' 121140  060000                    MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                        ; 1211
000671' 431020  000020
000672' 121200  001777                    MWORD   <CONT,A=2,B=12,SAQ,RMIN,D=2,SKCN,MGC=1777>  ; 1212
000673' 022025  240340
000674' 121340  060000                    MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                        ; 1213
```

```
595  000675' 431020  000020
596  000676' 121400  001777          MWORD   <CONT,A=3,B=13,SAB,OR,D=3,SKCN,MGC=1777>      ; 1214
597  000677' 133035  640340
598  000700' 121540  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                   ; 1215
599  000701' 431020  000020
600  000702' 121600  001777          MWORD   <CONT,A=4,B=14,SAQ,AND,D=4,SKCN,MGC=1777>     ; 1216
601  000703' 044046  240340
602  000704' 121740  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                   ; 1217
603  000705' 431020  000020
604  000706' 122000  001777          MWORD   <CONT,A=5,B=15,SAB,NAND,D=5,SKCN,MGC=1777>    ; 1220
605  000707' 155056  640340
606  000710' 122140  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                   ; 1221
607  000711' 431020  000020
608  000712' 122200  001777          MWORD   <CONT,A=6,B=16,SAQ,XOR,D=6,SKCN,MGC=1777>     ; 1222
609  000713' 066067  240340
610  000714' 122340  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                   ; 1223
611  000715' 431020  000020
612  000716' 122400  001777          MWORD   <CONT,A=7,B=17,SAB,XNOR,D=7,SKCN,MGC=1777>    ; 1224
613  000717' 177077  640340
614  000720' 122540  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                   ; 1225
615  000721' 431020  000020
616
617  000722' 122600  001777          MWORD   <CONT,A=2,B=10,SOQ,XNOR,D=0,SKCN,MGC=1777>    ; 1226
618  000723' 270024  240340
619  000724' 122740  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                   ; 1227
620  000725' 431020  000020
621  000726' 123000  001777          MWORD   <CONT,A=1,B=11,SOB,XOR,D=1,SKCN,MGC=1777>     ; 1230
622  000727' 361014  640340
623  000730' 123140  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                   ; 1231
624  000731' 431020  000020
625  000732' 123200  001777          MWORD   <CONT,A=2,B=12,SOQ,NAND,D=2,SKCN,MGC=1777>    ; 1232
626  000733' 252025  240340
627  000734' 123340  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                   ; 1233
628  000735' 431020  000020
629  000736' 123400  001777          MWORD   <CONT,A=3,B=13,SOB,AND,D=3,SKCN,MGC=1777>     ; 1234
630  000737' 343035  640340
631  000740' 123540  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                   ; 1235
632  000741' 431020  000020
633  000742' 123600  001777          MWORD   <CONT,A=4,B=14,SOQ,OR,D=4,SKCN,MGC=1777>      ; 1236
634  000743' 234046  240340
635  000744' 123740  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                   ; 1237
636  000745' 431020  000020
637  000746' 124000  001777          MWORD   <CONT,A=5,B=15,SOB,RMIN,D=5,SKCN,MGC=1777>    ; 1240
638  000747' 325056  640340
639  000750' 124140  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                   ; 1241
640  000751' 431020  000020
641  000752' 124200  001777          MWORD   <CONT,A=6,B=16,SOQ,SMIN,D=6,SKCN,MGC=1777>    ; 1242
642  000753' 216067  240340
643  000754' 124340  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                   ; 1243
644  000755' 431020  000020
645  000756' 124400  001777          MWORD   <CONT,A=7,B=17,SOB,PLUS,D=7,SKCN,MGC=1777>    ; 1244
646  000757' 307077  640340
647  000760' 124540  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                   ; 1245
648  000761' 431020  000020
649
```

```
 650  000762' 124600  001777          MWORD   <CONT,A=2,B=10,SOA,PLUS,D=0,SKCN,MGC=1777>  ; 1246
 651  000763' 400024  240340
 652  000764' 124740  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                 ; 1247
 653  000765' 431020  000020
 654  000766' 125000  001777          MWORD   <CONT,A=1,B=11,SDA,SMIN,D=1,SKCN,MGC=1777>  ; 1250
 655  000767' 511014  640340
 656  000770' 125140  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                 ; 1251
 657  000771' 431020  000020
 658  000772' 125200  001777          MWORD   <CONT,A=2,B=12,SOA,RMIN,D=2,SKCN,MGC=1777>  ; 1252
 659  000773' 422025  240340
 660  000774' 125340  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                 ; 1253
 661  000775' 431020  000020
 662  000776' 125400  001777          MWORD   <CONT,A=3,B=13,SDA,OR,D=3,SKCN,MGC=1777>    ; 1254
 663  000777' 533035  640340
 664  001000' 125540  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                 ; 1255
 665  001001' 431020  000020
 666  001002' 125600  001777          MWORD   <CONT,A=4,B=14,SOA,AND,D=4,SKCN,MGC=1777>   ; 1256
 667  001003' 444046  240340
 668  001004' 125740  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                 ; 1257
 669  001005' 431020  000020
 670  001006' 126000  001777          MWORD   <CONT,A=5,B=15,SDA,NAND,D=5,SKCN,MGC=1777>  ; 1260
 671  001007' 555056  640340
 672  001010' 126140  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                 ; 1261
 673  001011' 431020  000020
 674  001012' 126200  001777          MWORD   <CONT,A=6,B=16,SOA,XOR,D=6,SKCN,MGC=1777>   ; 1262
 675  001013' 466067  240340
 676  001014' 126340  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                 ; 1263
 677  001015' 431020  000020
 678  001016' 126400  001777          MWORD   <CONT,A=7,B=17,SDA,XNOR,D=7,SKCN,MGC=1777>  ; 1264
 679  001017' 577077  640340
 680  001020' 126540  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                 ; 1265
 681  001021' 431020  000020
 682
 683  001022' 126600  001777          MWORD   <CONT,A=2,B=10,SDQ,XNOR,D=0,SKCN,MGC=1777>  ; 1266
 684  001023' 670024  240340
 685  001024' 126740  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                 ; 1267
 686  001025' 431020  000020
 687  001026' 127000  001777          MWORD   <CONT,A=1,B=11,SDQ,XOR,D=1,SKCN,MGC=1777>   ; 1270
 688  001027' 761014  640340
 689  001030' 127140  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                 ; 1271
 690  001031' 431020  000020
 691  001032' 127200  001777          MWORD   <CONT,A=2,B=12,SDQ,NAND,D=2,SKCN,MGC=1777>  ; 1272
 692  001033' 652025  240340
 693  001034' 127340  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                 ; 1273
 694  001035' 431020  000020
 695  001036' 127400  001777          MWORD   <CONT,A=3,B=13,SDQ,AND,D=3,SKCN,MGC=1777>   ; 1274
 696  001037' 743035  640340
 697  001040' 127540  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                 ; 1275
 698  001041' 431020  000020
 699  001042' 127600  001777          MWORD   <CONT,A=4,B=14,SDQ,OR,D=4,SKCN,MGC=1777>    ; 1276
 700  001043' 634046  240340
 701  001044' 127740  060000          MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                 ; 1277
 702  001045' 431020  000020
 703  001046' 130000  001777          MWORD   <CONT,A=5,B=15,SDQ,RMIN.D=5,SKCN,MGC=1777>  ; 1300
 704  001047' 725056  640340
```

```
705  001050' 130140  060000           MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                         ; 1301
706  001051' 431020  000020
707  001052' 130200  001777           MWORD   <CONT,A=6,B=16,SDQ,SMIN,D=6,SKCN,MGC=1777>          ; 1302
708  001053' 616067  240340
709  001054' 130340  060000           MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                         ; 1303
710  001055' 431020  000020
711  001056' 130400  001777           MWORD   <CONT,A=7,B=17,SDO,PLUS,D=7,SKCN,MGC=1777>          ; 1304
712  001057' 707077  640340
713  001060' 130540  060000           MWORD   <CJS,J=4006,A=2,SOA,OR,D=1>                         ; 1305
714  001061' 431020  000020
715
716                             ;         Now do register 3
717
718  001062' 130600  020000           MWORD   <CJS,J=2>                                           ; 1306
719  001063' 000000  000020
720  001064' 130700  000000           MWORD   <CONT,A=0,B=3,SOA,AND,D=2>                          ; 1307
721  001065' 442001  400340
722
723  001066' 131000  001777           MWORD   <CONT,A=3,B=10,SAQ,PLUS,D=0,SKCN,MGC=1777>          ; 1310
724  001067' 000034  240340
725  001070' 131140  110000           MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                         ; 1311
726  001071' 431030  000020
727  001072' 131200  001777           MWORD   <CONT,A=1,B=11,SAB,SMIN,D=1,SKCN,MGC=1777>          ; 1312
728  001073' 111014  640340
729  001074' 131340  110000           MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                         ; 1313
730  001075' 431030  000020
731  001076' 131400  001777           MWORD   <CONT,A=2,B=12,SAQ,RMIN,D=2,SKCN,MGC=1777>          ; 1314
732  001077' 022025  240340
733  001100' 131540  110000           MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                         ; 1315
734  001101' 431030  000020
735  001102' 131600  001777           MWORD   <CONT,A=3,B=13,SAB,OR,D=3,SKCN,MGC=1777>            ; 1316
736  001103' 133035  640340
737  001104' 131740  110000           MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                         ; 1317
738  001105' 431030  000020
739  001106' 132000  001777           MWORD   <CONT,A=4,B=14,SAQ,AND,D=4,SKCN,MGC=1777>           ; 1320
740  001107' 044046  240340
741  001110' 132140  110000           MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                         ; 1321
742  001111' 431030  000020
743  001112' 132200  001777           MWORD   <CONT,A=5,B=15,SAB,NAND,D=5,SKCN,MGC=1777>          ; 1322
744  001113' 155056  640340
745  001114' 132340  110000           MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                         ; 1323
746  001115' 431030  000020
747  001116' 132400  001777           MWORD   <CONT,A=6,B=16,SAQ,XOR,D=6,SKCN,MGC=1777>           ; 1324
748  001117' 066067  240340
749  001120' 132540  110000           MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                         ; 1325
750  001121' 431030  000020
751  001122' 132600  001777           MWORD   <CONT,A=7,B=17,SAB,XNOR,D=7,SKCN,MGC=1777>          ; 1326
752  001123' 177077  640340
753  001124' 132740  110000           MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                         ; 1327
754  001125' 431030  000020
755
756  001126' 133000  001777           MWORD   <CONT,A=3,B=10,SOQ,XNOR,D=0,SKCN,MGC=1777>          ; 1330
757  001127' 270034  240340
758  001130' 133140  110000           MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                         ; 1331
759  001131' 431030  000020
```

```
001132' 133200  001777          MWORD   <CONT,A=1,B=11,SOB,XOR,D=1,SKCN,MGC=1777>     ; 1332
001133' 361014  640340
001134' 133340  110000          MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                   ; 1333
001135' 431030  000020
001136' 133400  001777          MWORD   <CONT,A=2,B=12,SOQ,NAND,D=2,SKCN,MGC=1777>    ; 1334
001137' 252025  240340
001140' 133540  110000          MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                   ; 1335
001141' 431030  000020
001142' 133600  001777          MWORD   <CONT,A=3,B=13,SOB,AND,D=3,SKCN,MGC=1777>     ; 1336
001143' 343035  640340
001144' 133740  110000          MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                   ; 1337
001145' 431030  000020
001146' 134000  001777          MWORD   <CONT,A=4,B=14,SOQ,OR,D=4,SKCN,MGC=1777>      ; 1340
001147' 234046  240340
001150' 134140  110000          MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                   ; 1341
001151' 431030  000020
001152' 134200  001777          MWORD   <CONT,A=5,B=15,SOB,RMIN,D=5,SKCN,MGC=1777>    ; 1342
001153' 325056  640340
001154' 134340  110000          MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                   ; 1343
001155' 431030  000020
001156' 134400  001777          MWORD   <CONT,A=6,B=16,SOQ,SMIN,D=6,SKCN,MGC=1777>    ; 1344
001157' 216067  240340
001160' 134540  110000          MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                   ; 1345
001161' 431030  000020
001162' 134600  001777          MWORD   <CONT,A=7,B=17,SOB,PLUS,D=7,SKCN,MGC=1777>    ; 1346
001163' 307077  640340
001164' 134740  110000          MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                   ; 1347
001165' 431030  000020

001166' 135000  001777          MWORD   <CONT,A=3,B=10,SOA,PLUS,D=0,SKCN,MGC=1777>    ; 1350
001167' 400034  240340
001170' 135140  110000          MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                   ; 1351
001171' 431030  000020
001172' 135200  001777          MWORD   <CONT,A=1,B=11,SDA,SMIN,D=1,SKCN,MGC=1777>    ; 1352
001173' 511014  640340
001174' 135340  110000          MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                   ; 1353
001175' 431030  000020
001176' 135400  001777          MWORD   <CONT,A=2,B=12,SOA,RMIN,D=2,SKCN,MGC=1777>    ; 1354
001177' 422025  240340
001200' 135540  110000          MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                   ; 1355
001201' 431030  000020
001202' 135600  001777          MWORD   <CONT,A=3,B=13,SDA,OR,D=3,SKCN,MGC=1777>      ; 1356
001203' 533035  640340
001204' 135740  110000          MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                   ; 1357
001205' 431030  000020
001206' 136000  001777          MWORD   <CONT,A=4,B=14,SOA,AND,D=4,SKCN,MGC=1777>     ; 1360
001207' 444046  240340
001210' 136140  110000          MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                   ; 1361
001211' 431030  000020
001212' 136200  001777          MWORD   <CONT,A=5,B=15,SDA,NAND,D=5,SKCN,MGC=1777>    ; 1362
001213' 555056  640340
001214' 136340  110000          MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                   ; 1363
001215' 431030  000020
001216' 136400  001777          MWORD   <CONT,A=6,B=16,SOA,XOR,D=6,SKCN,MGC=1777>     ; 1364
001217' 466067  240340
```

```
815  001220' 136540  110000                    MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                        ; 1365
816  001221' 431030  000020
817  001222' 136600  001777                    MWORD   <CONT,A=7,B=17,SDA,XNOR,D=7,SKCN,MGC=1777>  ; 1366
818  001223' 577077  640340
819  001224' 136740  110000                    MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                        ; 1367
820  001225' 431030  000020
821
822  001226' 137000  001777                    MWORD   <CONT,A=3,B=10,SDQ,XNOR,D=0,SKCN,MGC=1777>  ; 1370
823  001227' 670034  240340
824  001230' 137140  110000                    MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                        ; 1371
825  001231' 431030  000020
826  001232' 137200  001777                    MWORD   <CONT,A=1,B=11,SDO,XOR,D=1,SKCN,MGC=1777>   ; 1372
827  001233' 761014  640340
828  001234' 137340  110000                    MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                        ; 1373
829  001235' 431030  000020
830  001236' 137400  001777                    MWORD   <CONT,A=2,B=12,SDQ,NAND,D=2,SKCN,MGC=1777>  ; 1374
831  001237' 652025  240340
832  001240' 137540  110000                    MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                        ; 1375
833  001241' 431030  000020
834  001242' 137600  001777                    MWORD   <CONT,A=3,B=13,SDO,AND,D=3,SKCN,MGC=1777>   ; 1376
835  001243' 743035  640340
836  001244' 137740  110000                    MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                        ; 1377
837  001245' 431030  000020
838  001246' 140000  001777                    MWORD   <CONT,A=4,B=14,SDQ,OR,D=4,SKCN,MGC=1777>    ; 1400
839  001247' 634046  240340
840  001250' 140140  110000                    MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                        ; 1401
841  001251' 431030  000020
842  001252' 140200  001777                    MWORD   <CONT,A=5,B=15,SDO,RMIN,D=5,SKCN,MGC=1777>  ; 1402
843  001253' 725056  640340
844  001254' 140340  110000                    MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                        ; 1403
845  001255' 431030  000020
846  001256' 140400  001777                    MWORD   <CONT,A=6,B=16,SDQ,SMIN,D=6,SKCN,MGC=1777>  ; 1404
847  001257' 616067  240340
848  001260' 140540  110000                    MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                        ; 1405
849  001261' 431030  000020
850  001262' 140600  001777                    MWORD   <CONT,A=7,B=17,SDO,PLUS,D=7,SKCN,MGC=1777>  ; 1406
851  001263' 707077  640340
852  001264' 140740  110000                    MWORD   <CJS,J=4011,A=3,SOA,OR,D=1>                        ; 1407
853  001265' 431030  000020
854
855                                    ;       Now do register 4
856
857  001266' 141000  020000                    MWORD   <CJS,J=2>                                          ; 1410
858  001267' 000000  000020
859  001270' 141100  000000                    MWORD   <CONT,A=0,B=4,SOA,AND,D=2>                         ; 1411
860  001271' 442002  000340
861
862  001272' 141200  001777                    MWORD   <CONT,A=4,B=10,SAQ,PLUS,D=0,SKCN,MGC=1777>  ; 1412
863  001273' 000044  240340
864  001274' 141340  140000                    MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                        ; 1413
865  001275' 431040  000020
866  001276' 141400  001777                    MWORD   <CONT,A=1,B=11,SAB,SMIN,D=1,SKCN,MGC=1777>  ; 1414
867  001277' 111014  640340
868  001300' 141540  140000                    MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                        ; 1415
869  001301' 431040  000020
```

```
870  001302' 141600  001777          MWORD   <CONT,A=2,B=12,SAQ,RMIN,D=2,SKCN,MGC=1777>  ; 1416
871  001303' 022025  240340
872  001304' 141740  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1417
873  001305' 431040  000020
874  001306' 142000  001777          MWORD   <CONT,A=3,B=13,SAB,OR,D=3,SKCN,MGC=1777>    ; 1420
875  001307' 133035  640340
876  001310' 142140  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1421
877  001311' 431040  000020
878  001312' 142200  001777          MWORD   <CONT,A=4,B=14,SAQ,AND,D=4,SKCN,MGC=1777>   ; 1422
879  001313' 044046  240340
880  001314' 142340  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1423
881  001315' 431040  000020
882  001316' 142400  001777          MWORD   <CONT,A=5,B=15,SAB,NAND,D=5,SKCN,MGC=1777>  ; 1424
883  001317' 155056  640340
884  001320' 142540  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1425
885  001321' 431040  000020
886  001322' 142600  001777          MWORD   <CONT,A=6,B=16,SAQ,XOR,D=6,SKCN,MGC=1777>   ; 1426
887  001323' 066067  240340
888  001324' 142740  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1427
889  001325' 431040  000020
890  001326' 143000  001777          MWORD   <CONT,A=7,B=17,SAB,XNOR,D=7,SKCN,MGC=1777>  ; 1430
891  001327' 177077  640340
892  001330' 143140  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1431
893  001331' 431040  000020
894
895  001332' 143200  001777          MWORD   <CONT,A=4,B=10,SOQ,XNOR,D=0,SKCN,MGC=1777>  ; 1432
896  001333' 270044  240340
897  001334' 143340  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1433
898  001335' 431040  000020
899  001336' 143400  001777          MWORD   <CONT,A=1,B=11,SOB,XOR,D=1,SKCN,MGC=1777>   ; 1434
900  001337' 301014  640340
901  001340' 143540  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1435
902  001341' 431040  000020
903  001342' 143600  001777          MWORD   <CONT,A=2,B=12,SOQ,NAND,D=2,SKCN,MGC=1777>  ; 1436
904  001343' 252025  240340
905  001344' 143740  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1437
906  001345' 431040  000020
907  001346' 144000  001777          MWORD   <CONT,A=3,B=13,SOB,AND,D=3,SKCN,MGC=1777>   ; 1440
908  001347' 343035  640340
909  001350' 144140  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1441
910  001351' 431040  000020
911  001352' 144200  001777          MWORD   <CONT,A=4,B=14,SOQ,OR,D=4,SKCN,MGC=1777>    ; 1442
912  001353' 234046  240340
913  001354' 144340  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1443
914  001355' 431040  000020
915  001356' 144400  001777          MWORD   <CONT,A=5,B=15,SOB,RMIN,D=5,SKCN,MGC=1777>  ; 1444
916  001357' 325056  640340
917  001360' 144540  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1445
918  001361' 431040  000020
919  001362' 144600  001777          MWORD   <CONT,A=6,B=16,SOQ,SMIN,D=6,SKCN,MGC=1777>  ; 1446
920  001363' 216067  240340
921  001364' 144740  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1447
922  001365' 431040  000020
923  001366' 145000  001777          MWORD   <CONT,A=7,B=17,SOB,PLUS.D=7,SKCN,MGC=1777>  ; 1450
924  001367' 307077  640340
```

```
 925  001370' 145140  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                     ; 1451
 926  001371' 431040  000020
 927
 928  001372' 145200  001777          MWORD   <CONT,A=4,B=10,SOA,PLUS,D=0,SKCN,MGC=1777>      ; 1452
 929  001373' 400044  240340
 930  001374' 145340  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                     ; 1453
 931  001375' 431040  000020
 932  001376' 145400  001777          MWORD   <CONT,A=1,B=11,SDA,SMIN,D=1,SKCN,MGC=1777>      ; 1454
 933  001377' 511014  640340
 934  001400' 145540  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                     ; 1455
 935  001401' 431040  000020
 936  001402' 145600  001777          MWORD   <CONT,A=2,B=12,SOA,RMIN,D=2,SKCN,MGC=1777>      ; 1456
 937  001403' 422025  240340
 938  001404' 145740  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                     ; 1457
 939  001405' 431040  000020
 940  001406' 146000  001777          MWORD   <CONT,A=3,B=13,SDA,OR,D=3,SKCN,MGC=1777>        ; 1460
 941  001407' 533035  640340
 942  001410' 146140  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                     ; 1461
 943  001411' 431040  000020
 944  001412' 146200  001777          MWORD   <CONT,A=4,B=14,SOA,AND,D=4,SKCN,MGC=1777>       ; 1462
 945  001413' 444046  240340
 946  001414' 146340  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                     ; 1463
 947  001415' 431040  000020
 948  001416' 146400  001777          MWORD   <CONT,A=5,B=15,SDA,NAND,D=5,SKCN,MGC=1777>      ; 1464
 949  001417' 555056  640340
 950  001420' 146540  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                     ; 1465
 951  001421' 431040  000020
 952  001422' 146600  001777          MWORD   <CONT,A=6,B=16,SOA,XOR,D=6,SKCN,MGC=1777>       ; 1466
 953  001423' 466067  240340
 954  001424' 146740  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                     ; 1467
 955  001425' 431040  000020
 956  001426' 147000  001777          MWORD   <CONT,A=7,B=17,SDA,XNOR,D=7,SKCN,MGC=1777>      ; 1470
 957  001427' 577077  640340
 958  001430' 147140  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                     ; 1471
 959  001431' 431040  000020
 960
 961  001432' 147200  001777          MWORD   <CONT,A=4,B=10,SDQ,XNOR,D=0,SKCN,MGC=1777>      ; 1472
 962  001433' 670044  240340
 963  001434' 147340  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                     ; 1473
 964  001435' 431040  000020
 965  001436' 147400  001777          MWORD   <CONT,A=1,B=11,SDO,XOR,D=1,SKCN,MGC=1777>       ; 1474
 966  001437' 761014  640340
 967  001440' 147540  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                     ; 1475
 968  001441' 431040  000020
 969  001442' 147600  001777          MWORD   <CONT,A=2,B=12,SDQ,NAND,D=2,SKCN,MGC=1777>      ; 1476
 970  001443' 652025  240340
 971  001444' 147740  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                     ; 1477
 972  001445' 431040  000020
 973  001446' 150000  001777          MWORD   <CONT,A=3,B=13,SDO,AND,D=3,SKCN,MGC=1777>       ; 1500
 974  001447' 743035  640340
 975  001450' 150140  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                     ; 1501
 976  001451' 431040  000020
 977  001452' 150200  001777          MWORD   <CONT,A=4,B=14,SDQ,OR,D=4,SKCN,MGC=1777>        ; 1502
 978  001453' 634046  240340
 979  001454' 150340  140000          MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                     ; 1503
```

```
001455' 431040  000020
001456' 150400  001777                MWORD   <CONT,A=5,B=15,SDO,RMIN,D=5,SKCN,MGC=1777>  ; 1504
001457' 725056  640340
001460' 150540  140000                MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1505
001461' 431040  000020
001462' 150600  001777                MWORD   <CONT,A=6,B=16,SDQ,SMIN,D=6,SKCN,MGC=1777>  ; 1506
001463' 616067  240340
001464' 150740  140000                MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1507
001465' 431040  000020
001466' 151000  001777                MWORD   <CONT,A=7,B=17,SDO,PLUS,D=7,SKCN,MGC=1777>  ; 1510
001467' 707077  640340
001470' 151140  140000                MWORD   <CJS,J=4014,A=4,SOA,OR,D=1>                 ; 1511
001471' 431040  000020

                              ;       Now do register 5

001472' 151200  020000                MWORD   <CJS,J=2>                                   ; 1512
001473' 000000  000020
001474' 151300  000000                MWORD   <CONT,A=0,B=5,SOA,AND,D=2>                  ; 1513
001475' 442002  400340

001476' 151400  001777                MWORD   <CONT,A=5,B=10,SAQ,PLUS,D=0,SKCN,MGC=1777>  ; 1514
001477' 000054  240340
001500' 151540  170000                MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1515
001501' 431050  000020
001502' 151600  001777                MWORD   <CONT,A=1,B=11,SAB,SMIN,D=1,SKCN,MGC=1777>  ; 1516
001503' 111014  640340
001504  151740  170000                MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1517
001505' 431050  000020
001506' 152000  001777                MWORD   <CONT,A=2,B=12,SAQ,RMIN,D=2,SKCN,MGC=1777>  ; 1520
001507' 022025  240340
001510' 152140  170000                MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1521
001511' 431050  000020
001512' 152200  001777                MWORD   <CONT,A=3,B=13,SAB,OR,D=3,SKCN,MGC=1777>    ; 1522
001513' 133035  640340
001514' 152340  170000                MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1523
001515' 431050  000020
001516' 152400  001777                MWORD   <CONT,A=4,B=14,SAQ,AND,D=4,SKCN,MGC=1777>   ; 1524
001517' 044046  240340
001520' 152540  170000                MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1525
001521' 431050  000020
001522' 152600  001777                MWORD   <CONT,A=5,B=15,SAB,NAND,D=5,SKCN,MGC=1777>  ; 1526
001523' 155056  640340
001524' 152740  170000                MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1527
001525' 431050  000020
001526' 153000  001777                MWORD   <CONT,A=6,B=16,SAQ,XOR,D=6,SKCN,MGC=1777>   ; 1530
001527' 066067  240340
001530' 153140  170000                MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1531
001531' 431050  000020
001532' 153200  001777                MWORD   <CONT,A=7,B=17,SAB,XNOR,D=7,SKCN,MGC=1777>  ; 1532
001533' 177077  640340
001534' 153340  170000                MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1533
001535' 431050  000020

001536' 153400  001777                MWORD   <CONT,A=5,B=10,SOQ,XNOR,D=0,SKCN,MGC=1777>  ; 1534
```

```
001537' 270054  240340
001540' 153540  170000          MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                        ; 1535
001541' 431050  000020
001542' 153600  001777          MWORD   <CONT,A=1,B=11,SOB,XOR,D=1,SKCN,MGC=1777>          ; 1536
001543' 361014  640340
001544' 153740  170000          MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                        ; 1537
001545' 431050  000020
001546' 154000  001777          MWORD   <CONT,A=2,B=12,SOQ,NAND,D=2,SKCN,MGC=1777>         ; 1540
001547' 252025  240340
001550' 154140  170000          MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                        ; 1541
001551' 431050  000020
001552' 154200  001777          MWORD   <CONT,A=3,B=13,SOB,AND,D=3,SKCN,MGC=1777>          ; 1542
001553' 343035  640340
001554' 154340  170000          MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                        ; 1543
001555' 431050  000020
001556' 154400  001777          MWORD   <CONT,A=4,B=14,SOQ,OR,D=4,SKCN,MGC=1777>           ; 1544
001557' 234046  240340
001560' 154540  170000          MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                        ; 1545
001561' 431050  000020
001562' 154600  001777          MWORD   <CONT,A=5,B=15,SOB,RMIN,D=5,SKCN,MGC=1777>         ; 1546
001563' 325056  640340
001564' 154740  170000          MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                        ; 1547
001565' 431050  000020
001566' 155000  001777          MWORD   <CONT,A=6,B=16,SOQ,SMIN,D=6,SKCN,MGC=1777>         ; 1550
001567' 216067  240340
001570' 155140  170000          MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                        ; 1551
001571' 431050  000020
001572' 155200  001777          MWORD   <CONT,A=7,B=17,SOB,PLUS,D=7,SKCN,MGC=1777>         ; 1552
001573' 307077  640340
001574' 155340  170000          MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                        ; 1553
001575' 431050  000020

001576' 155400  001777          MWORD   <CONT,A=5,B=10,SOA,PLUS,D=0,SKCN,MGC=1777>         ; 1554
001577' 400054  240340
001600' 155540  170000          MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                        ; 1555
001601' 431050  000020
001602' 155600  001777          MWORD   <CONT,A=1,B=11,SDA,SMIN,D=1,SKCN,MGC=1777>         ; 1556
001603' 511014  640340
001604' 155740  170000          MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                        ; 1557
001605' 431050  000020
001606' 156000  001777          MWORD   <CONT,A=2,B=12,SOA,RMIN,D=2,SKCN,MGC=1777>         ; 1560
001607' 422025  240340
001610' 156140  170000          MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                        ; 1561
001611' 431050  000020
001612' 156200  001777          MWORD   <CONT,A=3,B=13,SDA,OR,D=3,SKCN,MGC=1777>           ; 1562
001613' 533035  640340
001614' 156340  170000          MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                        ; 1563
001615' 431050  000020
001616' 156400  001777          MWORD   <CONT,A=4,B=14,SOA,AND,D=4,SKCN,MGC=1777>          ; 1564
001617' 444046  240340
001620' 156540  170000          MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                        ; 1565
001621' 431050  000020
001622' 156600  001777          MWORD   <CONT,A=5,B=15,SDA,NAND,D=5,SKCN,MGC=1777>         ; 1566
001623' 555056  640340
001624' 156740  170000          MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                        ; 1567
```

```
001625' 431050  000020
001626' 157000  001777              MWORD   <CONT,A=6,B=16,SOA,XOR,D=6,SKCN,MGC=1777>   ; 1570
001627' 466067  240340
001630' 157140  170000              MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1571
001631' 431050  000020
001632' 157200  001777              MWORD   <CONT,A=7,B=17,SDA,XNOR,D=7,SKCN,MGC=1777>  ; 1572
001633' 577077  640340
001634' 157340  170000              MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1573
001635' 431050  000020

001636' 157400  001777              MWORD   <CONT,A=5,B=10,SDQ,XNOR,D=0,SKCN,MGC=1777>  ; 1574
001637' 670054  240340
001640' 157540  170000              MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1575
001641' 431050  000020
001642' 157600  001777              MWORD   <CONT,A=1,B=11,SDO,XOR,D=1,SKCN,MGC=1777>   ; 1576
001643' 761014  640340
001644' 157740  170000              MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1577
001645' 431050  000020
001646' 160000  001777              MWORD   <CONT,A=2,B=12,SDQ,NAND,D=2,SKCN,MGC=1777>  ; 1600
001647' 652025  240340
001650' 160140  170000              MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1601
001651' 431050  000020
001652' 160200  001777              MWORD   <CONT,A=3,B=13,SDO,AND,D=3,SKCN,MGC=1777>   ; 1602
001653' 743035  640340
001654' 160340  170000              MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1603
001655' 431050  000020
001656' 160400  001777              MWORD   <CONT,A=4,B=14,SDQ,OR,D=4,SKCN,MGC=1777>    ; 1604
001657' 634046  240340
001660' 160540  170000              MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1605
001661' 431050  000020
001662' 160600  001777              MWORD   <CONT,A=5,B=15,SDO,RMIN,D=5,SKCN,MGC=1777>  ; 1606
001663' 725056  640340
001664' 160740  170000              MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1607
001665' 431050  000020
001666' 161000  001777              MWORD   <CONT,A=6,B=16,SDQ,SMIN,D=6,SKCN,MGC=1777>  ; 1610
001667' 616067  240340
001670' 161140  170000              MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1611
001671' 431050  000020
001672' 161200  001777              MWORD   <CONT,A=7,B=17,SDO,PLUS,D=7,SKCN,MGC=1777>  ; 1612
001673' 707077  640340
001674' 161340  170000              MWORD   <CJS,J=4017,A=5,SOA,OR,D=1>                 ; 1613
001675' 431050  000020


                            ;       Now do register 6

001676' 161400  020000              MWORD   <CJS,J=2>                                   ; 1614
001677' 000000  000020
001700' 161500  000000              MWORD   <CONT,A=0,B=6,SOA,AND,D=2>                  ; 1615
001701' 442003  000340

001702' 161600  001777              MWORD   <CONT,A=6,B=10,SAQ,PLUS,D=0,SKCN,MGC=1777>  ; 1616
001703' 000064  240340
001704' 161740  220000              MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                 ; 1617
001705' 431060  000020
001706' 162000  001777              MWORD   <CONT,A=1,B=11,SAB,SMIN,D=1,SKCN,MGC=1777>  ; 1620
```

```
 1145  001707' 111014  640340
 1146  001710' 162140  220000                MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                              ; 1621
 1147  001711' 431060  000020
 1148  001712' 162200  001777                MWORD   <CONT,A=2,B=12,SAQ,RMIN,D=2,SKCN,MGC=1777>               ; 1622
 1149  001713' 022025  240340
 1150  001714' 162340  220000                MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                              ; 1623
 1151  001715' 431060  000020
 1152  001716' 162400  001777                MWORD   <CONT,A=3,B=13,SAB,OR,D=3,SKCN,MGC=1777>                 ; 1624
 1153  001717' 133035  640340
 1154  001720' 162540  220000                MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                              ; 1625
 1155  001721' 431060  000020
 1156  001722' 162600  001777                MWORD   <CONT,A=4,B=14,SAQ,AND,D=4,SKCN,MGC=1777>                ; 1626
 1157  001723' 044046  240340
 1158  001724' 162740  220000                MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                              ; 1627
 1159  001725' 431060  000020
 1160  001726' 163000  001777                MWORD   <CONT,A=5,B=15,SAB,NAND,D=5,SKCN,MGC=1777>               ; 1630
 1161  001727' 155056  640340
 1162  001730' 163140  220000                MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                              ; 1631
 1163  001731' 431060  000020
 1164  001732' 163200  001777                MWORD   <CONT,A=6,B=16,SAQ,XOR,D=6,SKCN,MGC=1777>                ; 1632
 1165  001733' 066067  240340
 1166  001734' 163340  220000                MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                              ; 1633
 1167  001735' 431060  000020
 1168  001736' 163400  001777                MWORD   <CONT,A=7,B=17,SAB,XNOR,D=7,SKCN,MGC=1777>               ; 1634
 1169  001737' 177077  640340
 1170  001740' 163540  220000                MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                              ; 1635
 1171  001741' 431060  000020
 1172
 1173  001742' 163600  001777                MWORD   <CONT,A=6,B=10,SOQ,XNOR,D=0,SKCN,MGC=1777>               ; 1636
 1174  001743' 270064  240340
 1175  001744' 163740  220000                MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                              ; 1637
 1176  001745' 431060  000020
 1177  001746' 164000  001777                MWORD   <CONT,A=1,B=11,SOB,XOR,D=1,SKCN,MGC=1777>                ; 1640
 1178  001747' 361014  640340
 1179  001750' 164140  220000                MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                              ; 1641
 1180  001751' 431060  000020
 1181  001752' 164200  001777                MWORD   <CONT,A=2,B=12,SOQ,NAND,D=2,SKCN,MGC=1777>               ; 1642
 1182  001753' 252025  240340
 1183  001754' 164340  220000                MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                              ; 1643
 1184  001755' 431060  000020
 1185  001756' 164400  001777                MWORD   <CONT,A=3,B=13,SOB,AND,D=3,SKCN,MGC=1777>                ; 1644
 1186  001757' 343035  640340
 1187  001760' 164540  220000                MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                              ; 1645
 1188  001761' 431060  000020
 1189  001762' 164600  001777                MWORD   <CONT,A=4,B=14,SOQ,OR,D=4,SKCN,MGC=1777>                 ; 1646
 1190  001763' 234046  240340
 1191  001764' 164740  220000                MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                              ; 1647
 1192  001765' 431060  000020
 1193  001766' 165000  001777                MWORD   <CONT,A=5,B=15,SOB,RMIN,D=5,SKCN,MGC=1777>               ; 1650
 1194  001767' 325056  640340
 1195  001770' 165140  220000                MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                              ; 1651
 1196  001771' 431060  000020
 1197  001772' 165200  001777                MWORD   <CONT,A=6,B=16,SOQ,SMIN,D=6,SKCN,MGC=1777>               ; 1652
 1198  001773' 216067  240340
 1199  001774' 165340  220000                MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                              ; 1653
```

```
001775' 431060  000020
001776' 165400  001777          MWORD   <CONT,A=7,B=17,SOB,PLUS,D=7,SKCN,MGC=1777>  ; 1654
001777' 307077  640340
002000' 165540  220000          MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                  ; 1655
002001' 431060  000020

002002' 165600  001777          MWORD   <CONT,A=6,B=10,SOA,PLUS,D=0,SKCN,MGC=1777>  ; 1656
002003' 400064  240340
002004' 165740  220000          MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                  ; 1657
002005' 431060  000020
002006' 166000  001777          MWORD   <CONT,A=1,B=11,SDA,SMIN,D=1,SKCN,MGC=1777>  ; 1660
002007' 511014  640340
002010' 166140  220000          MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                  ; 1661
002011' 431060  000020
002012' 166200  001777          MWORD   <CONT,A=2,B=12,SOA,RMIN,D=2,SKCN,MGC=1777>  ; 1662
002013' 422025  240340
002014' 166340  220000          MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                  ; 1663
002015' 431060  000020
002016' 166400  001777          MWORD   <CONT,A=3,B=13,SDA,OR,D=3,SKCN,MGC=1777>    ; 1664
002017' 533035  640340
002020' 166540  220000          MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                  ; 1665
002021' 431060  000020
002022' 166600  001777          MWORD   <CONT,A=4,B=14,SOA,AND,D=4,SKCN,MGC=1777>   ; 1666
002023' 444046  240340
002024' 166740  220000          MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                  ; 1667
002025' 431060  000020
002026' 167000  001777          MWORD   <CONT,A=5,B=15,SDA,NAND,D=5,SKCN,MGC=1777>  ; 1670
002027' 555056  640340
002030' 167140  220000          MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                  ; 1671
002031' 431060  000020
002032' 167200  001777          MWORD   <CONT,A=6,B=16,SOA,XOR,D=6,SKCN,MGC=1777>   ; 1672
002033' 466067  240340
002034' 167340  220000          MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                  ; 1673
002035' 431060  000020
002036' 167400  001777          MWORD   <CONT,A=7,B=17,SDA,XNOR,D=7,SKCN,MGC=1777>  ; 1674
002037' 577077  640340
002040' 167540  220000          MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                  ; 1675
002041' 431060  000020

002042' 167600  001777          MWORD   <CONT,A=6,B=10,SDQ,XNOR,D=0,SKCN,MGC=1777>  ; 1676
002043' 670064  240340
002044' 167740  220000          MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                  ; 1677
002045' 431060  000020
002046' 170000  001777          MWORD   <CONT,A=1,B=11,SDO,XOR,D=1,SKCN,MGC=1777>   ; 1700
002047' 761014  640340
002050' 170140  220000          MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                  ; 1701
002051' 431060  000020
002052' 170200  001777          MWORD   <CONT,A=2,B=12,SDQ,NAND,D=2,SKCN,MGC=1777>  ; 1702
[illegible]53' 652025  240340
[illegible]054' 170340  220000          MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                  ; 1703
002055' 431060  000020
002056' 170400  001777          MWORD   <CONT,A=3,B=13,SDO,AND,D=3,SKCN,MGC=1777>   ; 1704
002057' 743035  640340
002060' 170540  220000          MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                  ; 1705
002061' 431060  000020
```

```
002062' 170600  001777                  MWORD   <CONT,A=4,B=14,SDQ,OR,D=4,SKCN,MGC=1777>    ; 1706
002063' 634046  240340
002064' 170740  220000                  MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                 ; 1707
002065' 431060  000020
002066' 171000  001777                  MWORD   <CONT,A=5,B=15,SDO,RMIN,D=5,SKCN,MGC=1777>  ; 1710
002067' 725056  640340
002070' 171140  220000                  MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                 ; 1711
002071' 431060  000020
002072' 171200  001777                  MWORD   <CONT,A=6,B=16,SDQ,SMIN,D=6,SKCN,MGC=1777>  ; 1712
002073' 616067  240340
002074' 171340  220000                  MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                 ; 1713
002075' 431060  000020
002076' 171400  001777                  MWORD   <CONT,A=7,B=17,SDO,PLUS,D=7,SKCN,MGC=1777>  ; 1714
002077' 707077  640340
002100' 171540  220000                  MWORD   <CJS,J=4022,A=6,SOA,OR,D=1>                 ; 1715
002101' 431060  000020

                                ;       Now do register 7

002102' 171600  020000                  MWORD   <CJS,J=2>                                   ; 1716
002103' 000000  000020
002104' 171700  000000                  MWORD   <CONT,A=0,B=7,SOA,AND,D=2>                  ; 1717
002105' 442003  400340

002106' 172000  001777                  MWORD   <CONT,A=7,B=10,SAQ,PLUS,D=0,SKCN,MGC=1777>  ; 1720
002107' 000074  240340
002110' 172140  250000                  MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1721
002111' 431070  000020
002112' 172200  001777                  MWORD   <CONT,A=1,B=11,SAB,SMIN,D=1,SKCN,MGC=1777>  ; 1722
002113' 111014  640340
002114' 172340  250000                  MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1723
002115' 431070  000020
002116' 172400  001777                  MWORD   <CONT,A=2,B=12,SAQ,RMIN,D=2,SKCN,MGC=1777>  ; 1724
002117' 022025  240340
002120' 172540  250000                  MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1725
002121' 431070  000020
002122' 172600  001777                  MWORD   <CONT,A=3,B=13,SAB,OR,D=3,SKCN,MGC=1777>    ; 1726
002123' 133035  640340
002124' 172740  250000                  MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1727
002125' 431070  000020
002126' 173000  001777                  MWORD   <CONT,A=4,B=14,SAQ,AND,D=4,SKCN,MGC=1777>   ; 1730
002127' 044046  240340
002130' 173140  250000                  MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1731
002131' 431070  000020
002132' 173200  001777                  MWORD   <CONT,A=5,B=15,SAB,NAND,D=5,SKCN,MGC=1777>  ; 1732
002133' 155056  640340
002134' 173340  250000                  MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1733
002135' 431070  000020
002136' 173400  001777                  MWORD   <CONT,A=6,B=16,SAQ,XOR,D=6,SKCN,MGC=1777>   ; 1734
002137' 066067  240340
002140' 173540  250000                  MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1735
002141' 431070  000020
002142' 173600  001777                  MWORD   <CONT,A=7,B=17,SAB,XNOR,D=7,SKCN,MGC=1777>  ; 1736
002143' 177077  640340
002144' 173740  250000                  MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1737
```

```
002145' 431070  000020

002146' 174000  001777          MWORD   <CONT,A=7,B=10,SOQ,XNOR,D=0,SKCN,MGC=1777>  ; 1740
002147' 270074  240340
002150' 174140  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1741
002151' 431070  000020
002152' 174200  001777          MWORD   <CONT,A=1,B=11,SOB,XOR,D=1,SKCN,MGC=1777>   ; 1742
002153' 361014  640340
002154' 174340  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1743
002155' 431070  000020
002156' 174400  001777          MWORD   <CONT,A=2,B=12,SOQ,NAND,D=2,SKCN,MGC=1777>  ; 1744
002157' 252025  240340
002160' 174540  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1745
002161' 431070  000020
002162' 174600  001777          MWORD   <CONT,A=3,B=13,SOB,AND,D=3,SKCN,MGC=1777>   ; 1746
002163' 343035  640340
002164' 174740  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1747
002165' 431070  000020
002166' 175000  001777          MWORD   <CONT,A=4,B=14,SOQ,OR,D=4,SKCN,MGC=1777>    ; 1750
002167' 234046  240340
002170' 175140  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1751
002171' 431070  000020
002172' 175200  001777          MWORD   <CONT,A=5,B=15,SOB,RMIN,D=5,SKCN,MGC=1777>  ; 1752
002173' 325056  640340
002174' 175340  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1753
002175' 431070  000020
002176' 175400  001777          MWORD   <CONT,A=6,B=16,SOQ,SMIN,D=6,SKCN,MGC=1777>  ; 1754
002177' 216067  240340
002200' 175540  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1755
002201' 431070  000020
002202' 175600  001777          MWORD   <CONT,A=7,B=17,SOB,PLUS,D=7,SKCN,MGC=1777>  ; 1756
002203' 307077  640340
002204' 175740  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1757
002205' 431070  000020

002206' 176000  001777          MWORD   <CONT,A=7,B=10,SOA,PLUS,D=0,SKCN,MGC=1777>  ; 1760
002207' 400074  240340
002210' 176140  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1761
002211' 431070  000020
002212' 176200  001777          MWORD   <CONT,A=1,B=11,SDA,SMIN,D=1,SKCN,MGC=1777>  ; 1762
002213' 511014  640340
002214' 176340  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1763
002215' 431070  000020
002216' 176400  001777          MWORD   <CONT,A=2,B=12,SOA,RMIN,D=2,SKCN,MGC=1777>  ; 1764
002217' 422025  240340
002220' 176540  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1765
002221' 431070  000020
002222' 176600  001777          MWORD   <CONT,A=3,B=13,SDA,OR,D=3,SKCN,MGC=1777>    ; 1766
002223' 533035  640340
002224' 176740  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1767
002225' 431070  000020
002226' 177000  001777          MWORD   <CONT,A=4,B=14,SOA,AND,D=4,SKCN,MGC=1777>   ; 1770
002227' 444046  240340
002230' 177140  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1771
002231' 431070  000020
```

```
002232' 177200  001777          MWORD   <CONT,A=5,B=15,SDA,NAND,D=5,SKCN,MGC=1777>  ; 1772
002233' 555056  640340
002234' 177340  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1773
002235' 431070  000020
002236' 177400  001777          MWORD   <CONT,A=6,B=16,SOA,XOR,D=6,SKCN,MGC=1777>   ; 1774
002237' 466067  240340
002240' 177540  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1775
002241' 431070  000020
002242' 177600  001777          MWORD   <CONT,A=7,B=17,SDA,XNOR,D=7,SKCN,MGC=1777>  ; 1776
002243' 577077  640340
002244' 177740  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 1777
002245' 431070  000020

002246' 200000  001777          MWORD   <CONT,A=7,B=10,SDQ,XNOR,D=0,SKCN,MGC=1777>  ; 2000
002247' 670074  240340
002250' 200140  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 2001
002251' 431070  000020
002252' 200200  001777          MWORD   <CONT,A=1,B=11,SDO,XOR,D=1,SKCN,MGC=1777>   ; 2002
002253' 761014  640340
002254' 200340  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 2003
002255' 431070  000020
002256' 200400  001777          MWORD   <CONT,A=2,B=12,SDQ,NAND,D=2,SKCN,MGC=1777>  ; 2004
002257' 652025  240340
002260' 200540  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 2005
002261' 431070  000020
002262' 200600  001777          MWORD   <CONT,A=3,B=13,SDO,AND,D=3,SKCN,MGC=1777>   ; 2006
002263' 743035  640340
002264' 200740  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 2007
002265' 431070  000020
002266' 201000  001777          MWORD   <CONT,A=4,B=14,SDQ,OR,D=4,SKCN,MGC=1777>    ; 2010
002267' 634046  240340
002270' 201140  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 2011
002271' 431070  000020
002272' 201200  001777          MWORD   <CONT,A=5,B=15,SDO,RMIN,D=5,SKCN,MGC=1777>  ; 2012
002273' 725056  640340
002274' 201340  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 2013
002275' 431070  000020
002276' 201400  001777          MWORD   <CONT,A=6,B=16,SDQ,SMIN,D=6,SKCN,MGC=1777>  ; 2014
002277' 616067  240340
002300' 201540  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 2015
002301' 431070  000020
002302' 201600  001777          MWORD   <CONT,A=7,B=17,SDO,PLUS,D=7,SKCN,MGC=1777>  ; 2016
002303' 707077  640340
002304' 201740  250000          MWORD   <CJS,J=4025,A=7,SOA,OR,D=1>                 ; 2017
002305' 431070  000020

                        ;       Now do register 10

002306' 202000  020000          MWORD   <CJS,J=2>                                   ; 2020
002307' 000000  000020
002310' 202100  000000          MWORD   <CONT,A=0,B=10,SOA,AND,D=2>                 ; 2021
002311' 442004  000340

002312' 202200  001777          MWORD   <CONT,A=10,B=10,SAQ,PLUS,D=0,SKCN,MGC=1777> ; 2022
002313' 000104  240340
```

```
002314' 202340  300000         MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2023
002315' 431100  000020
002316' 202400  001777         MWORD   <CONT,A=1,B=11,SAB,SMIN,D=1,SKCN,MGC=1777>      ; 2024
002317' 111014  640340
002320' 202540  300000         MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2025
002321' 431100  000020
002322' 202600  001777         MWORD   <CONT,A=2,B=12,SAQ,RMIN,D=2,SKCN,MGC=1777>      ; 2026
002323' 022025  240340
002324' 202740  300000         MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2027
002325' 431100  000020
002326' 203000  001777         MWORD   <CONT,A=3,B=13,SAB,OR,D=3,SKCN,MGC=1777>        ; 2030
002327' 133035  640340
002330' 203140  300000         MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2031
002331' 431100  000020
002332' 203200  001777         MWORD   <CONT,A=4,B=14,SAQ,AND,D=4,SKCN,MGC=1777>       ; 2032
002333' 044046  240340
002334' 203340  300000         MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2033
002335' 431100  000020
002336' 203400  001777         MWORD   <CONT,A=5,B=15,SAB,NAND,D=5,SKCN,MGC=1777>      ; 2034
002337' 155056  640340
002340' 203540  300000         MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2035
002341' 431100  000020
002342' 203600  001777         MWORD   <CONT,A=6,B=16,SAQ,XOR,D=6,SKCN,MGC=1777>       ; 2036
002343' 066067  240340
002344' 203740  300000         MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2037
002345' 431100  000020
002346' 204000  001777         MWORD   <CONT,A=7,B=17,SAB,XNOR,D=7,SKCN,MGC=1777>      ; 2040
002347' 177077  640340
002350' 204140  300000         MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2041
002351' 431100  000020

002352' 204200  001777         MWORD   <CONT,A=10,B=10,SOQ,XNOR,D=0,SKCN,MGC=1777>     ; 2042
002353' 270104  240340
002354' 204340  300000         MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2043
002355' 431100  000020
002356' 204400  001777         MWORD   <CONT,A=1,B=11,SOB,XOR,D=1,SKCN,MGC=1777>       ; 2044
002357' 361014  640340
002360' 204540  300000         MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2045
002361' 431100  000020
002362' 204600  001777         MWORD   <CONT,A=2,B=12,SOQ,NAND,D=2,SKCN,MGC=1777>      ; 2046
002363' 252025  240340
002364' 204740  300000         MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2047
002365' 431100  000020
002366' 205000  001777         MWORD   <CONT,A=3,B=13,SOB,AND,D=3,SKCN,MGC=1777>       ; 2050
002367' 343035  640340
002370' 205140  300000         MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2051
002371' 431100  000020
002372' 205200  001777         MWORD   <CONT,A=4,B=14,SOQ,OR,D=4,SKCN,MGC=1777>        ; 2052
002373' 234046  240340
002374' 205340  300000         MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2053
002375' 431100  000020
002376' 205400  001777         MWORD   <CONT,A=5,B=15,SOB,RMIN,D=5,SKCN,MGC=1777>      ; 2054
002377' 325056  640340
002400' 205540  300000         MWORD   <CJS,J=4030,A=10,SOA,OR.D=1>                    ; 2055
002401' 431100  000020
```

```
 1475  002402' 205600  001777           MWORD   <CONT,A=6,B=16,SOQ,SMIN,D=6,SKCN,MGC=1777>  ; 2056
 1476  002403' 216067  240340
 1477  002404' 205740  300000           MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                ; 2057
 1478  002405' 431100  000020
 1479  002406' 206000  001777           MWORD   <CONT,A=7,B=17,SOB,PLUS,D=7,SKCN,MGC=1777>  ; 2060
 1480  002407' 307077  640340
 1481  002410' 206140  300000           MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                ; 2061
 1482  002411' 431100  000020
 1483
 1484  002412' 206200  001777           MWORD   <CONT,A=10,B=10,SOA,PLUS,D=0,SKCN,MGC=1777> ; 2062
 1485  002413' 400104  240340
 1486  002414' 206340  300000           MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                ; 2063
 1487  002415' 431100  000020
 1488  002416' 206400  001777           MWORD   <CONT,A=1,B=11,SDA,SMIN,D=1,SKCN,MGC=1777>  ; 2064
 1489  002417' 511014  640340
 1490  002420' 206540  300000           MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                ; 2065
 1491  002421' 431100  000020
 1492  002422' 206600  001777           MWORD   <CONT,A=2,B=12,SOA,RMIN,D=2,SKCN,MGC=1777>  ; 2066
 1493  002423' 422025  240340
 1494  002424' 206740  300000           MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                ; 2067
 1495  002425' 431100  000020
 1496  002426' 207000  001777           MWORD   <CONT,A=3,B=13,SDA,OR,D=3,SKCN,MGC=1777>    ; 2070
 1497  002427' 533035  640340
 1498  002430' 207140  300000           MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                ; 2071
 1499  002431' 431100  000020
 1500  002432' 207200  001777           MWORD   <CONT,A=4,B=14,SOA,AND,D=4,SKCN,MGC=1777>   ; 2072
 1501  002433' 444046  240340
 1502  002434' 207340  300000           MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                ; 2073
 1503  002435' 431100  000020
 1504  002436' 207400  001777           MWORD   <CONT,A=5,B=15,SDA,NAND,D=5,SKCN,MGC=1777>  ; 2074
 1505  002437' 555056  640340
 1506  002440' 207540  300000           MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                ; 2075
 1507  002441' 431100  000020
 1508  002442' 207600  001777           MWORD   <CONT,A=6,B=16,SOA,XOR,D=6,SKCN,MGC=1777>   ; 2076
 1509  002443' 466067  240340
 1510  002444' 207740  300000           MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                ; 2077
 1511  002445' 431100  000020
 1512  002446' 210000  001777           MWORD   <CONT,A=7,B=17,SDA,XNOR,D=7,SKCN,MGC=1777>  ; 2100
 1513  002447' 577077  640340
 1514  002450' 210140  300000           MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                ; 2101
 1515  002451' 431100  000020
 1516
 1517  002452' 210200  001777           MWORD   <CONT,A=10,B=10,SDQ,XNOR,D=0,SKCN,MGC=1777> ; 2102
 1518  002453' 670104  240340
 1519  002454' 210340  300000           MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                ; 2103
 1520  002455' 431100  000020
 1521  002456' 210400  001777           MWORD   <CONT,A=1,B=11,SDO,XOR,D=1,SKCN,MGC=1777>   ; 2104
 1522  002457' 761014  640340
 1523  002460' 210540  300000           MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                ; 2105
 1524  002461' 431100  000020
 1525  002462' 210600  001777           MWORD   <CONT,A=2,B=12,SDQ,NAND,D=2,SKCN,MGC=1777>  ; 2106
 1526  002463' 652025  240340
 1527  002464' 210740  300000           MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                ; 2107
 1528  002465' 431100  000020
 1529  002466' 211000  001777           MWORD   <CONT,A=3,B=13,SDO,AND,D=3,SKCN,MGC=1777>   ; 2110
```

```
002467' 743035  640340
002470' 211140  300000            MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2111
002471' 431100  000020
002472' 211200  001777            MWORD   <CONT,A=4,B=14,SDQ,OR,D=4,SKCN,MGC=1777>        ; 2112
002473' 634046  240340
002474' 211340  300000            MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2113
002475' 431100  000020
002476' 211400  001777            MWORD   <CONT,A=5,B=15,SDO,RMIN,D=5,SKCN,MGC=1777>      ; 2114
002477' 725056  640340
002500' 211540  300000            MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2115
002501' 431100  000020
002502' 211600  001777            MWORD   <CONT,A=6,B=16,SDQ,SMIN,D=6,SKCN,MGC=1777>      ; 2116
002503' 616067  240340
002504' 211740  300000            MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2117
002505' 431100  000020
002506' 212000  001777            MWORD   <CONT,A=7,B=17,SDO,PLUS,D=7,SKCN,MGC=1777>      ; 2120
002507' 707077  640340
002510' 212140  300000            MWORD   <CJS,J=4030,A=10,SOA,OR,D=1>                    ; 2121
002511' 431100  000020


                          ;       Now do register 11

002512' 212200  020000            MWORD   <CJS,J=2>                                       ; 2122
002513' 000000  000020
002514' 212300  000000            MWORD   <CONT,A=0,B=11,SOA,AND,D=2>                     ; 2123
002515' 442004  400340

002516' 212400  001777            MWORD   <CONT,A=10,B=0,SAQ,PLUS,D=0,SKCN,MGC=1777>      ; 2124
002517' 000100  240340
002520' 212540  330000            MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2125
002521' 431110  000020
002522' 212600  001777            MWORD   <CONT,A=11,B=1,SAB,SMIN,D=1,SKCN,MGC=1777>      ; 2126
002523' 111110  640340
002524' 212740  330000            MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2127
002525' 431110  000020
002526' 213000  001777            MWORD   <CONT,A=12,B=2,SAQ,RMIN,D=2,SKCN,MGC=1777>      ; 2130
002527' 022121  240340
002530' 213140  330000            MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2131
002531' 431110  000020
002532' 213200  001777            MWORD   <CONT,A=13,B=3,SAB,OR,D=3,SKCN,MGC=1777>        ; 2132
002533' 133131  640340
002534' 213340  330000            MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2133
002535' 431110  000020
002536' 213400  001777            MWORD   <CONT,A=14,B=4,SAQ,AND,D=4,SKCN,MGC=1777>       ; 2134
002537' 044142  240340
002540' 213540  330000            MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2135
002541' 431110  000020
002542' 213600  001777            MWORD   <CONT,A=15,B=5,SAB,NAND,D=5,SKCN,MGC=1777>      ; 2136
002543' 155152  640340
002544' 213740  330000            MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2137
002545' 431110  000020
002546' 214000  001777            MWORD   <CONT,A=16,B=6,SAQ,XOR,D=6,SKCN,MGC=1777>       ; 2140
002547' 066163  240340
002550' 214140  330000            MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2141
002551' 431110  000020
```

```
 1585  002552' 214200  001777         MWORD   <CONT,A=17,B=7,SAB,XNOR,D=7,SKCN,MGC=1777>  ; 2142
 1586  002553' 177173  640340
 1587  002554' 214340  330000         MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                ; 2143
 1588  002555' 431110  000020
 1589
 1590  002556' 214400  001777         MWORD   <CONT,A=10,B=0,SOQ,XNOR,D=0,SKCN,MGC=1777>  ; 2144
 1591  002557' 270100  240340
 1592  002560' 214540  330000         MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                ; 2145
 1593  002561' 431110  000020
 1594  002562' 214600  001777         MWORD   <CONT,A=11,B=1,SOB,XOR,D=1,SKCN,MGC=1777>   ; 2146
 1595  002563' 361110  640340
 1596  002564' 214740  330000         MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                ; 2147
 1597  002565' 431110  000020
 1598  002566' 215000  001777         MWORD   <CONT,A=12,B=2,SOQ,NAND,D=2,SKCN,MGC=1777>  ; 2150
 1599  002567' 252121  240340
 1600  002570' 215140  330000         MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                ; 2151
 1601  002571' 431110  000020
 1602  002572' 215200  001777         MWORD   <CONT,A=13,B=3,SOB,AND,D=3,SKCN,MGC=1777>   ; 2152
 1603  002573' 343131  640340
 1604  002574' 215340  330000         MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                ; 2153
 1605  002575' 431110  000020
 1606  002576' 215400  001777         MWORD   <CONT,A=14,B=4,SOQ,OR,D=4,SKCN,MGC=1777>    ; 2154
 1607  002577' 234142  240340
 1608  002600' 215540  330000         MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                ; 2155
 1609  002601' 431110  000020
 1610  002602' 215600  001777         MWORD   <CONT,A=15,B=5,SOB,RMIN,D=5,SKCN,MGC=1777>  ; 2156
 1611  002603' 325152  640340
 1612  002604' 215740  330000         MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                ; 2157
 1613  002605' 431110  000020
 1614  002606' 216000  001777         MWORD   <CONT,A=16,B=6,SOQ,SMIN,D=6,SKCN,MGC=1777>  ; 2160
 1615  002607' 216163  240340
 1616  002610' 216140  330000         MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                ; 2161
 1617  002611' 431110  000020
 1618  002612' 216200  001777         MWORD   <CONT,A=17,B=7,SOB,PLUS,D=7,SKCN,MGC=1777>  ; 2162
 1619  002613' 307173  640340
 1620  002614' 216340  330000         MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                ; 2163
 1621  002615' 431110  000020
 1622
 1623  002616' 216400  001777         MWORD   <CONT,A=10,B=0,SOA,PLUS,D=0,SKCN,MGC=1777>  ; 2164
 1624  002617' 400100  240340
 1625  002620' 216540  330000         MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                ; 2165
 1626  002621' 431110  000020
 1627  002622' 216600  001777         MWORD   <CONT,A=11,B=1,SDA,SMIN,D=1,SKCN,MGC=1777>  ; 2166
 1628  002623' 511110  640340
 1629  002624' 216740  330000         MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                ; 2167
 1630  002625' 431110  000020
 1631  002626' 217000  001777         MWORD   <CONT,A=12,B=2,SOA,RMIN,D=2,SKCN,MGC=1777>  ; 2170
 1632  002627' 422121  240340
 1633  002630' 217140  330000         MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                ; 2171
 1634  002631' 431110  000020
 1635  002632' 217200  001777         MWORD   <CONT,A=13,B=3,SDA,OR,D=3,SKCN,MGC=1777>    ; 2172
 1636  002633' 533131  640340
 1637  002634' 217340  330000         MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                ; 2173
 1638  002635' 431110  000020
 1639  002636' 217400  001777         MWORD   <CONT,A=14,B=4,SOA,AND,D=4,SKCN,MGC=1777>   ; 2174
```

```
002637' 444142  240340
002640' 217540  330000          MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2175
002641' 431110  000020
002642' 217600  001777          MWORD   <CONT,A=15,B=5,SDA,NAND,D=5,SKCN,MGC=1777>      ; 2176
002643' 555152  640340
002644' 217740  330000          MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2177
002645' 431110  000020
002646' 220000  001777          MWORD   <CONT,A=16,B=6,SOA,XOR,D=6,SKCN,MGC=1777>       ; 2200
002647' 466163  240340
002650' 220140  330000          MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2201
002651' 431110  000020
002652' 220200  001777          MWORD   <CONT,A=17,B=7,SDA,XNOR,D=7,SKCN,MGC=1777>      ; 2202
002653' 577173  640340
002654' 220340  330000          MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2203
002655' 431110  000020

002656' 220400  001777          MWORD   <CONT,A=10,B=0,SDQ,XNOR,D=0,SKCN,MGC=1777>      ; 2204
002657' 670100  240340
002660' 220540  330000          MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2205
002661' 431110  000020
002662' 220600  001777          MWORD   <CONT,A=11,B=1,SDO,XOR,D=1,SKCN,MGC=1777>       ; 2206
002663' 761110  640340
002664' 220740  330000          MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2207
002665' 431110  000020
002666' 221000  001777          MWORD   <CONT,A=12,B=2,SDQ,NAND,D=2,SKCN,MGC=1777>      ; 2210
002667' 652121  240340
002670' 221140  330000          MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2211
002671' 431110  000020
002672' 221200  001777          MWORD   <CONT,A=13,B=3,SDO,AND,D=3,SKCN,MGC=1777>       ; 2212
002673' 743131  640340
002674' 221340  330000          MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2213
002675' 431110  000020
002676' 221400  001777          MWORD   <CONT,A=14,B=4,SDQ,OR,D=4,SKCN,MGC=1777>        ; 2214
002677' 634142  240340
002700' 221540  330000          MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2215
002701' 431110  000020
002702' 221600  001777          MWORD   <CONT,A=15,B=5,SDO,RMIN,D=5,SKCN,MGC=1777>      ; 2216
002703' 725152  640340
002704' 221740  330000          MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2217
002705' 431110  000020
002706' 222000  001777          MWORD   <CONT,A=16,B=6,SDQ,SMIN,D=6,SKCN,MGC=1777>      ; 2220
002707' 616163  240340
002710' 222140  330000          MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2221
002711' 431110  000020
002712' 222200  001777          MWORD   <CONT,A=17,B=7,SDO,PLUS,D=7,SKCN,MGC=1777>      ; 2222
002713' 707173  640340
002714' 222340  330000          MWORD   <CJS,J=4033,A=11,SOA,OR,D=1>                    ; 2223
002715' 431110  000020

                        ;       Now do register 12

002716' 222400  020000          MWORD   <CJS,J=2>                                       ; 2224
002717' 000000  000020
002720' 222500  000000          MWORD   <CONT,A=0,B=12,SOA,AND,D=2>                     ; 2225
002721' 442005  000340
```

```
1695
1696  002722' 222600  001777                  MWORD   <CONT,A=10,B=0,SAQ,PLUS,D=0,SKCN,MGC=1777>  ; 2226
1697  002723' 000100  240340
1698  002724' 222740  360000                  MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>               ; 2227
1699  002725' 431120  000020
1700  002726' 223000  001777                  MWORD   <CONT,A=11,B=1,SAB,SMIN,D=1,SKCN,MGC=1777>  ; 2230
1701  002727' 111110  640340
1702  002730' 223140  360000                  MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>               ; 2231
1703  002731' 431120  000020
1704  002732' 223200  001777                  MWORD   <CONT,A=12,B=2,SAQ,RMIN,D=2,SKCN,MGC=1777>  ; 2232
1705  002733' 022121  240340
1706  002734' 223340  360000                  MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>               ; 2233
1707  002735' 431120  000020
1708  002736' 223400  001777                  MWORD   <CONT,A=13,B=3,SAB,OR,D=3,SKCN,MGC=1777>    ; 2234
1709  002737' 133131  640340
1710  002740' 223540  360000                  MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>               ; 2235
1711  002741' 431120  000020
1712  002742' 223600  001777                  MWORD   <CONT,A=14,B=4,SAQ,AND,D=4,SKCN,MGC=1777>   ; 2236
1713  002743' 044142  240340
1714  002744' 223740  360000                  MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>               ; 2237
1715  002745' 431120  000020
1716  002746' 224000  001777                  MWORD   <CONT,A=15,B=5,SAB,NAND,D=5,SKCN,MGC=1777>  ; 2240
1717  002747' 155152  640340
1718  002750' 224140  360000                  MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>               ; 2241
1719  002751' 431120  000020
1720  002752' 224200  001777                  MWORD   <CONT,A=16,B=6,SAQ,XOR,D=6,SKCN,MGC=1777>   ; 2242
1721  002753' 066163  240340
1722  002754' 224340  360000                  MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>               ; 2243
1723  002755' 431120  000020
1724  002756' 224400  001777                  MWORD   <CONT,A=17,B=7,SAB,XNOR,D=7,SKCN,MGC=1777>  ; 2244
1725  002757' 177173  640340
1726  002760' 224540  360000                  MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>               ; 2245
1727  002761' 431120  000020
1728
1729  002762' 224600  001777                  MWORD   <CONT,A=10,B=0,SOQ,XNOR,D=0,SKCN,MGC=1777>  ; 2246
1730  002763' 270100  240340
1731  002764' 224740  360000                  MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>               ; 2247
1732  002765' 431120  000020
1733  002766' 225000  001777                  MWORD   <CONT,A=11,B=1,SOB,XOR,D=1,SKCN,MGC=1777>   ; 2250
1734  002767' 361110  640340
1735  002770' 225140  360000                  MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>               ; 2251
1736  002771' 431120  000020
1737  002772' 225200  001777                  MWORD   <CONT,A=12,B=2,SOQ,NAND,D=2,SKCN,MGC=1777>  ; 2252
1738  002773' 252121  240340
1739  002774' 225340  360000                  MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>               ; 2253
1740  002775' 431120  000020
1741  002776' 225400  001777                  MWORD   <CONT,A=13,B=3,SOB,AND,D=3,SKCN,MGC=1777>   ; 2254
1742  002777' 343131  640340
1743  003000' 225540  360000                  MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>               ; 2255
1744  003001' 431120  000020
1745  003002' 225600  001777                  MWORD   <CONT,A=14,B=4,SOQ,OR,D=4,SKCN,MGC=1777>    ; 2256
1746  003003' 234142  240340
1747  003004' 225740  360000                  MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>               ; 2257
1748  003005' 431120  000020
1749  003006' 226000  001777                  MWORD   <CONT,A=15,B=5,SOB,RMIN,D=5,SKCN,MGC=1777>  ; 2260
```

```
1750  003007' 325152  640340
1751  003010' 226140  360000          MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                        ; 2261
1752  003011' 431120  000020
1753  003012' 226200  001777          MWORD   <CONT,A=16,B=6,SOQ,SMIN,D=6,SKCN,MGC=1777>          ; 2262
1754  003013' 216163  240340
1755  003014' 226340  360000          MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                        ; 2263
1756  003015' 431120  000020
1757  003016' 226400  001777          MWORD   <CONT,A=17,B=7,SOB,PLUS,D=7,SKCN,MGC=1777>          ; 2264
1758  003017' 307173  640340
1759  003020' 226540  360000          MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                        ; 2265
1760  003021' 431120  000020
1761
1762  003022' 226600  001777          MWORD   <CONT,A=10,B=0,SOA,PLUS,D=0,SKCN,MGC=1777>          ; 2266
1763  003023' 400100  240340
1764  003024' 226740  360000          MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                        ; 2267
1765  003025' 431120  000020
1766  003026' 227000  001777          MWORD   <CONT,A=11,B=1,SDA,SMIN,D=1,SKCN,MGC=1777>          ; 2270
1767  003027' 511110  640340
1768  003030' 227140  360000          MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                        ; 2271
1769  003031' 431120  000020
1770  003032' 227200  001777          MWORD   <CONT,A=12,B=2,SOA,RMIN,D=2,SKCN,MGC=1777>          ; 2272
1771  003033' 422121  240340
1772  003034' 227340  360000          MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                        ; 2273
1773  003035' 431120  000020
1774  003036' 227400  001777          MWORD   <CONT,A=13,B=3,SDA,OR,D=3,SKCN,MGC=1777>            ; 2274
1775  003037' 533131  640340
1776  003040' 227540  360000          MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                        ; 2275
1777  003041' 431120  000020
1778  003042' 227600  001777          MWORD   <CONT,A=14,B=4,SOA,AND,D=4,SKCN,MGC=1777>           ; 2276
1779  003043' 444142  240340
1780  003044' 227740  360000          MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                        ; 2277
1781  003045' 431120  000020
1782  003046' 230000  001777          MWORD   <CONT,A=15,B=5,SDA,NAND,D=5,SKCN,MGC=1777>          ; 2300
1783  003047' 555152  640340
1784  003050' 230140  360000          MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                        ; 2301
1785  003051' 431120  000020
1786  003052' 230200  001777          MWORD   <CONT,A=16,B=6,SOA,XOR,D=6,SKCN,MGC=1777>           ; 2302
1787  003053' 466163  240340
1788  003054' 230340  360000          MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                        ; 2303
1789  003055' 431120  000020
1790  003056' 230400  001777          MWORD   <CONT,A=17,B=7,SDA,XNOR,D=7,SKCN,MGC=1777>          ; 2304
1791  003057' 577173  640340
1792  003060' 230540  360000          MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                        ; 2305
1793  003061' 431120  000020
1794
1795  003062' 230600  001777          MWORD   <CONT,A=10,B=0,SDQ,XNOR,D=0,SKCN,MGC=1777>          ; 2306
1796  003063' 670100  240340
1797  003064' 230740  360000          MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                        ; 2307
1798  003065' 431120  000020
1799  003066' 231000  001777          MWORD   <CONT,A=11,B=1,SDO,XOR,D=1,SKCN,MGC=1777>           ; 2310
1800  003067' 761110  640340
1801  003070' 231140  360000          MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                        ; 2311
1802  003071' 431120  000020
1803  003072' 231200  001777          MWORD   <CONT,A=12,B=2,SDQ,NAND.D=2,SKCN,MGC=1777>          ; 2312
1804  003073' 652121  240340
```

```
003074' 231340  360000                    MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                          ; 2313
003075' 431120  000020
003076' 231400  001777                    MWORD   <CONT,A=13,B=3,SDO,AND,D=3,SKCN,MGC=1777>             ; 2314
003077' 743131  640340
003100' 231540  360000                    MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                          ; 2315
003101' 431120  000020
003102' 231600  001777                    MWORD   <CONT,A=14,B=4,SDQ,OR,D=4,SKCN,MGC=1777>              ; 2316
003103' 634142  240340
003104' 231740  360000                    MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                          ; 2317
003105' 431120  000020
003106' 232000  001777                    MWORD   <CONT,A=15,B=5,SDO,RMIN,D=5,SKCN,MGC=1777>            ; 2320
003107' 725152  640340
003110' 232140  360000                    MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                          ; 2321
003111' 431120  000020
003112' 232200  001777                    MWORD   <CONT,A=16,B=6,SDQ,SMIN,D=6,SKCN,MGC=1777>            ; 2322
003113' 616163  240340
003114' 232340  360000                    MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                          ; 2323
003115' 431120  000020
003116' 232400  001777                    MWORD   <CONT,A=17,B=7,SDO,PLUS,D=7,SKCN,MGC=1777>            ; 2324
003117' 707173  640340
003120' 232540  360000                    MWORD   <CJS,J=4036,A=12,SOA,OR,D=1>                          ; 2325
003121' 431120  000020

                                  ;       Now do register 13

003122' 232600  020000                    MWORD   <CJS,J=2>                                             ; 2326
003123' 000000  000020
003124' 232700  000000                    MWORD   <CONT,A=0,B=13,SOA,AND,D=2>                           ; 2327
003125' 442005  400340

003126' 233000  001777                    MWORD   <CONT,A=10,B=0,SAQ,PLUS,D=0,SKCN,MGC=1777>            ; 2330
003127' 000100  240340
003130' 233140  410000                    MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2331
003131' 431130  000020
003132' 233200  001777                    MWORD   <CONT,A=11,B=1,SAB,SMIN,D=1,SKCN,MGC=1777>            ; 2332
003133' 111110  640340
003134' 233340  410000                    MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2333
003135' 431130  000020
003136' 233400  001777                    MWORD   <CONT,A=12,B=2,SAQ,RMIN,D=2,SKCN,MGC=1777>            ; 2334
003137' 022121  240340
003140' 233540  410000                    MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2335
003141' 431130  000020
003142' 233600  001777                    MWORD   <CONT,A=13,B=3,SAB,OR,D=3,SKCN,MGC=1777>              ; 2336
003143' 133131  640340
003144' 233740  410000                    MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2337
003145' 431130  000020
003146' 234000  001777                    MWORD   <CONT,A=14,B=4,SAQ,AND,D=4,SKCN,MGC=1777>             ; 2340
003147' 044142  240340
003150' 234140  410000                    MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2341
003151' 431130  000020
003152' 234200  001777                    MWORD   <CONT,A=15,B=5,SAB,NAND,D=5,SKCN,MGC=1777>            ; 2342
003153' 155152  640340
003154' 234340  410000                    MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2343
003155' 431130  000020
003156' 234400  001777                    MWORD   <CONT,A=16,B=6,SAQ,XOR,D=6,SKCN,MGC=1777>             ; 2344
```

```
 1860  003157' 066163  240340
 1861  003160' 234540  410000                  MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2345
 1862  003161' 431130  000020
 1863  003162' 234600  001777                  MWORD   <CONT,A=17,B=7,SAB,XNOR,D=7,SKCN,MGC=1777>   ; 2346
 1864  003163' 177173  640340
 1865  003164' 234740  410000                  MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2347
 1866  003165' 431130  000020
 1867
 1868  003166' 235000  001777                  MWORD   <CONT,A=10,B=0,SOQ,XNOR,D=0,SKCN,MGC=1777>   ; 2350
 1869  003167' 270100  240340
 1870  003170' 235140  410000                  MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2351
 1871  003171' 431130  000020
 1872  003172' 235200  001777                  MWORD   <CONT,A=11,B=1,SOB,XOR,D=1,SKCN,MGC=1777>    ; 2352
 1873  003173' 361110  640340
 1874  003174' 235340  410000                  MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2353
 1875  003175' 431130  000020
 1876  003176' 235400  001777                  MWORD   <CONT,A=12,B=2,SOQ,NAND,D=2,SKCN,MGC=1777>   ; 2354
 1877  003177' 252121  240340
 1878  003200' 235540  410000                  MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2355
 1879  003201' 431130  000020
 1880  003202' 235600  001777                  MWORD   <CONT,A=13,B=3,SOB,AND,D=3,SKCN,MGC=1777>    ; 2356
 1881  003203' 343131  640340
 1882  003204' 235740  410000                  MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2357
 1883  003205' 431130  000020
 1884  003206' 236000  001777                  MWORD   <CONT,A=14,B=4,SOQ,OR,D=4,SKCN,MGC=1777>     ; 2360
 1885  003207' 234142  240340
 1886  003210' 236140  410000                  MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2361
 1887  003211' 431130  000020
 1888  003212' 236200  001777                  MWORD   <CONT,A=15,B=5,SOB,RMIN,D=5,SKCN,MGC=1777>   ; 2362
 1889  003213' 325152  640340
 1890  003214' 236340  410000                  MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2363
 1891  003215' 431130  000020
 1892  003216' 236400  001777                  MWORD   <CONT,A=16,B=6,SOQ,SMIN,D=6,SKCN,MGC=1777>   ; 2364
 1893  003217' 216163  240340
 1894  003220' 236540  410000                  MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2365
 1895  003221' 431130  000020
 1896  003222' 236600  001777                  MWORD   <CONT,A=17,B=7,SOB,PLUS,D=7,SKCN,MGC=1777>   ; 2366
 1897  003223' 307173  640340
 1898  003224' 236740  410000                  MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2367
 1899  003225' 431130  000020
 1900
 1901  003226' 237000  001777                  MWORD   <CONT,A=10,B=0,SOA,PLUS,D=0,SKCN,MGC=1777>   ; 2370
 1902  003227' 400100  240340
 1903  003230' 237140  410000                  MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2371
 1904  003231' 431130  000020
 1905  003232' 237200  001777                  MWORD   <CONT,A=11,B=1,SDA,SMIN,D=1,SKCN,MGC=1777>   ; 2372
 1906  003233' 511110  640340
 1907  003234' 237340  410000                  MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2373
 1908  003235' 431130  000020
 1909  003236' 237400  001777                  MWORD   <CONT,A=12,B=2,SOA,RMIN,D=2,SKCN,MGC=1777>   ; 2374
 1910  003237' 422121  240340
 1911  003240' 237540  410000                  MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                          ; 2375
 1912  003241' 431130  000020
 1913  003242' 237600  001777                  MWORD   <CONT,A=13,B=3,SDA,OR,D=3,SKCN,MGC=1777>     ; 2376
 1914  003243' 533131  640340
```

```
1915  003244' 237740  410000            MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                    ; 2377
1916  003245' 431130  000020
1917  003246' 240000  001777            MWORD   <CONT,A=14,B=4,SOA,AND,D=4,SKCN,MGC=1777>       ; 2400
1918  003247' 444142  240340
1919  003250' 240140  410000            MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                    ; 2401
1920  003251' 431130  000020
1921  003252' 240200  001777            MWORD   <CONT,A=15,B=5,SDA,NAND,D=5,SKCN,MGC=1777>      ; 2402
1922  003253' 555152  640340
1923  003254' 240340  410000            MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                    ; 2403
1924  003255' 431130  000020
1925  003256' 240400  001777            MWORD   <CONT,A=16,B=6,SOA,XOR,D=6,SKCN,MGC=1777>       ; 2404
1926  003257' 466163  240340
1927  003260' 240540  410000            MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                    ; 2405
1928  003261' 431130  000020
1929  003262' 240600  001777            MWORD   <CONT,A=17,B=7,SDA,XNOR,D=7,SKCN,MGC=1777>      ; 2406
1930  003263' 577173  640340
1931  003264' 240740  410000            MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                    ; 2407
1932  003265' 431130  000020
1933
1934  003266' 241000  001777            MWORD   <CONT,A=10,B=0,SDQ,XNOR,D=0,SKCN,MGC=1777>      ; 2410
1935  003267' 670100  240340
1936  003270' 241140  410000            MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                    ; 2411
1937  003271' 431130  000020
1938  003272' 241200  001777            MWORD   <CONT,A=11,B=1,SDO,XOR,D=1,SKCN,MGC=1777>       ; 2412
1939  003273' 761110  640340
1940  003274' 241340  410000            MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                    ; 2413
1941  003275' 431130  000020
1942  003276' 241400  001777            MWORD   <CONT,A=12,B=2,SDQ,NAND,D=2,SKCN,MGC=1777>      ; 2414
1943  003277' 652121  240340
1944  003300' 241540  410000            MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                    ; 2415
1945  003301' 431130  000020
1946  003302' 241600  001777            MWORD   <CONT,A=13,B=3,SDO,AND,D=3,SKCN,MGC=1777>       ; 2416
1947  003303' 743131  640340
1948  003304' 241740  410000            MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                    ; 2417
1949  003305' 431130  000020
1950  003306' 242000  001777            MWORD   <CONT,A=14,B=4,SDQ,OR,D=4,SKCN,MGC=1777>        ; 2420
1951  003307' 634142  240340
1952  003310' 242140  410000            MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                    ; 2421
1953  003311' 431130  000020
1954  003312' 242200  001777            MWORD   <CONT,A=15,B=5,SDO,RMIN,D=5,SKCN,MGC=1777>      ; 2422
1955  003313' 725152  640340
1956  003314' 242340  410000            MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                    ; 2423
1957  003315' 431130  000020
1958  003316' 242400  001777            MWORD   <CONT,A=16,B=6,SDQ,SMIN,D=6,SKCN,MGC=1777>      ; 2424
1959  003317' 616163  240340
1960  003320' 242540  410000            MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                    ; 2425
1961  003321' 431130  000020
1962  003322' 242600  001777            MWORD   <CONT,A=17,B=7,SDO,PLUS,D=7,SKCN,MGC=1777>      ; 2426
1963  003323' 707173  640340
1964  003324' 242740  410000            MWORD   <CJS,J=4041,A=13,SOA,OR,D=1>                    ; 2427
1965  003325' 431130  000020
1966
1967                            ;       Now do register 14
1968
1969  003326' 243000  020000            MWORD   <CJS,J=2>                                       ; 2430
```

```
 1970  003327' 000000  000020
 1971  003330' 243100  000000                  MWORD   <CONT,A=0,B=14,SOA,AND,D=2>                         ; 2431
 1972  003331' 442006  000340
 1973
 1974  003332' 243200  001777                  MWORD   <CONT,A=10,B=0,SAQ,PLUS,D=0,SKCN,MGC=1777>          ; 2432
 1975  003333' 000100  240340
 1976  003334' 243340  440000                  MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                        ; 2433
 1977  003335' 431140  000020
 1978  003336' 243400  001777                  MWORD   <CONT,A=11,B=1,SAB,SMIN,D=1,SKCN,MGC=1777>          ; 2434
 1979  003337' 111110  640340
 1980  003340' 243540  440000                  MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                        ; 2435
 1981  003341' 431140  000020
 1982  003342' 243600  001777                  MWORD   <CONT,A=12,B=2,SAQ,RMIN,D=2,SKCN,MGC=1777>          ; 2436
 1983  003343' 022121  240340
 1984  003344' 243740  440000                  MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                        ; 2437
 1985  003345' 431140  000020
 1986  003346' 244000  001777                  MWORD   <CONT,A=13,B=3,SAB,OR,D=3,SKCN,MGC=1777>            ; 2440
 1987  003347' 133131  640340
 1988  003350' 244140  440000                  MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                        ; 2441
 1989  003351' 431140  000020
 1990  003352' 244200  001777                  MWORD   <CONT,A=14,B=4,SAQ,AND,D=4,SKCN,MGC=1777>           ; 2442
 1991  003353' 044142  240340
 1992  003354' 244340  440000                  MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                        ; 2443
 1993  003355' 431140  000020
 1994  003356' 244400  001777                  MWORD   <CONT,A=15,B=5,SAB,NAND,D=5,SKCN,MGC=1777>          ; 2444
 1995  003357' 155152  640340
 1996  003360' 244540  440000                  MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                        ; 2445
 1997  003361' 431140  000020
 1998  003362' 244600  001777                  MWORD   <CONT,A=16,B=6,SAQ,XOR,D=6,SKCN,MGC=1777>           ; 2446
 1999  003363' 066163  240340
 2000  003364' 244740  440000                  MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                        ; 2447
 2001  003365' 431140  000020
 2002  003366' 245000  001777                  MWORD   <CONT,A=17,B=7,SAB,XNOR,D=7,SKCN,MGC=1777>          ; 2450
 2003  003367' 177173  640340
 2004  003370' 245140  440000                  MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                        ; 2451
 2005  003371' 431140  000020
 2006
 2007  003372' 245200  001777                  MWORD   <CONT,A=10,B=0,SOQ,XNOR,D=0,SKCN,MGC=1777>          ; 2452
 2008  003373' 270100  240340
 2009  003374' 245340  440000                  MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                        ; 2453
 2010  003375' 431140  000020
 2011  003376' 245400  001777                  MWORD   <CONT,A=11,B=1,SOB,XOR,D=1,SKCN,MGC=1777>           ; 2454
 2012  003377' 361110  640340
 2013  003400' 245540  440000                  MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                        ; 2455
 2014  003401' 431140  000020
 2015  003402' 245600  001777                  MWORD   <CONT,A=12,B=2,SOQ,NAND,D=2,SKCN,MGC=1777>          ; 2456
 2016  003403' 252121  240340
 2017  003404' 245740  440000                  MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                        ; 2457
 2018  003405' 431140  000020
 2019  003406' 246000  001777                  MWORD   <CONT,A=13,B=3,SOB,AND,D=3,SKCN,MGC=1777>           ; 2460
 2020  003407' 343131  640340
 2021  003410' 246140  440000                  MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                        ; 2461
 2022  003411' 431140  000020
 2023  003412' 246200  001777                  MWORD   <CONT,A=14,B=4,SOQ,OR,D=4,SKCN,MGC=1777>            ; 2462
 2024  003413' 234142  240340
```

```
003414' 246340  440000              MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                      ; 2463
003415' 431140  000020
003416' 246400  001777              MWORD   <CONT,A=15,B=5,SOB,RMIN,D=5,SKCN,MGC=1777>        ; 2464
003417' 325152  640340
003420' 246540  440000              MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                      ; 2465
003421' 431140  000020
003422' 246600  001777              MWORD   <CONT,A=16,B=6,SOQ,SMIN,D=6,SKCN,MGC=1777>        ; 2466
003423' 216163  240340
003424' 246740  440000              MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                      ; 2467
003425' 431140  000020
003426' 247000  001777              MWORD   <CONT,A=17,B=7,SOB,PLUS,D=7,SKCN,MGC=1777>        ; 2470
003427' 307173  640340
003430' 247140  440000              MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                      ; 2471
003431' 431140  000020

003432' 247200  001777              MWORD   <CONT,A=10,B=0,SOA,PLUS,D=0,SKCN,MGC=1777>        ; 2472
003433' 400100  240340
003434' 247340  440000              MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                      ; 2473
003435' 431140  000020
003436' 247400  001777              MWORD   <CONT,A=11,B=1,SDA,SMIN,D=1,SKCN,MGC=1777>        ; 2474
003437' 511110  640340
003440' 247540  440000              MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                      ; 2475
003441' 431140  000020
003442' 247600  001777              MWORD   <CONT,A=12,B=2,SOA,RMIN,D=2,SKCN,MGC=1777>        ; 2476
003443' 422121  240340
003444' 247740  440000              MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                      ; 2477
003445' 431140  000020
003446' 250000  001777              MWORD   <CONT,A=13,B=3,SDA,OR,D=3,SKCN,MGC=1777>          ; 2500
003447' 533131  640340
003450' 250140  440000              MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                      ; 2501
003451' 431140  000020
003452' 250200  001777              MWORD   <CONT,A=14,B=4,SOA,AND,D=4,SKCN,MGC=1777>         ; 2502
003453' 444142  240340
003454' 250340  440000              MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                      ; 2503
003455' 431140  000020
003456' 250400  001777              MWORD   <CONT,A=15,B=5,SDA,NAND,D=5,SKCN,MGC=1777>        ; 2504
003457' 555152  640340
003460' 250540  440000              MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                      ; 2505
003461' 431140  000020
003462' 250600  001777              MWORD   <CONT,A=16,B=6,SOA,XOR,D=6,SKCN,MGC=1777>         ; 2506
003463' 466163  240340
003464' 250740  440000              MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                      ; 2507
003465' 431140  000020
003466' 251000  001777              MWORD   <CONT,A=17,B=7,SDA,XNOR,D=7,SKCN,MGC=1777>        ; 2510
003467' 577173  640340
003470' 251140  440000              MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                      ; 2511
003471' 431140  000020

003472' 251200  001777              MWORD   <CONT,A=10,B=0,SDQ,XNOR,D=0,SKCN,MGC=1777>        ; 2512
003473' 670100  240340
003474' 251340  440000              MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                      ; 2513
003475' 431140  000020
003476' 251400  001777              MWORD   <CONT,A=11,B=1,SDO,XOR,D=1,SKCN,MGC=1777>         ; 2514
003477' 761110  640340
003500' 251540  440000              MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                      ; 2515
```

```
003501' 431140  000020
003502' 251600  001777            MWORD   <CONT,A=12,B=2,SDQ,NAND,D=2,SKCN,MGC=1777>  ; 2516
003503' 652121  240340
003504' 251740  440000            MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                ; 2517
003505' 431140  000020
003506' 252000  001777            MWORD   <CONT,A=13,B=3,SDO,AND,D=3,SKCN,MGC=1777>   ; 2520
003507' 743131  640340
003510' 252140  440000            MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                ; 2521
003511' 431140  000020
003512' 252200  001777            MWORD   <CONT,A=14,B=4,SDQ,OR,D=4,SKCN,MGC=1777>    ; 2522
003513' 634142  240340
003514' 252340  440000            MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                ; 2523
003515' 431140  000020
003516' 252400  001777            MWORD   <CONT,A=15,B=5,SDO,RMIN,D=5,SKCN,MGC=1777>  ; 2524
003517' 725152  640340
003520' 252540  440000            MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                ; 2525
003521' 431140  000020
003522' 252600  001777            MWORD   <CONT,A=16,B=6,SDQ,SMIN,D=6,SKCN,MGC=1777>  ; 2526
003523' 616163  240340
003524' 252740  440000            MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                ; 2527
003525' 431140  000020
003526' 253000  001777            MWORD   <CONT,A=17,B=7,SDO,PLUS,D=7,SKCN,MGC=1777>  ; 2530
003527' 707173  640340
003530' 253140  440000            MWORD   <CJS,J=4044,A=14,SOA,OR,D=1>                ; 2531
003531' 431140  000020

                          ;       Now do register 15

003532' 253200  020000            MWORD   <CJS,J=2>                                   ; 2532
003533' 000000  000020
003534' 253300  000000            MWORD   <CONT,A=0,B=15,SOA,AND,D=2>                 ; 2533
003535' 442006  400340

003536' 253400  001777            MWORD   <CONT,A=10,B=0,SAQ,PLUS,D=0,SKCN,MGC=1777>  ; 2534
003537' 000100  240340
003540' 253540  470000            MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2535
003541' 431150  000020
003542' 253600  001777            MWORD   <CONT,A=11,B=1,SAB,SMIN,D=1,SKCN,MGC=1777>  ; 2536
003543' 111110  640340
003544' 253740  470000            MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2537
003545' 431150  000020
003546' 254000  001777            MWORD   <CONT,A=12,B=2,SAQ,RMIN,D=2,SKCN,MGC=1777>  ; 2540
003547' 022121  240340
003550' 254140  470000            MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2541
003551' 431150  000020
003552' 254200  001777            MWORD   <CONT,A=13,B=3,SAB,OR,D=3,SKCN,MGC=1777>    ; 2542
003553' 133131  640340
003554' 254340  470000            MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2543
003555' 431150  000020
003556' 254400  001777            MWORD   <CONT,A=14,B=4,SAQ,AND,D=4,SKCN,MGC=1777>   ; 2544
003557' 044142  240340
003560' 254540  470000            MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2545
003561' 431150  000020
003562' 254600  001777            MWORD   <CONT,A=15,B=5,SAB,NAND,D=5,SKCN,MGC=1777>  ; 2546
003563' 155152  640340
```

$B

```
 2135  003564' 254740  470000              MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                      ; 2547
 2136  003565' 431150  000020
 2137  003566' 255000  001777              MWORD   <CONT,A=16,B=6,SAQ,XOR,D=6,SKCN,MGC=1777>         ; 2550
 2138  003567' 066163  240340
 2139  003570' 255140  470000              MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                      ; 2551
 2140  003571' 431150  000020
 2141  003572' 255200  001777              MWORD   <CONT,A=17,B=7,SAB,XNOR,D=7,SKCN,MGC=1777>        ; 2552
 2142  003573' 177173  640340
 2143  003574' 255340  470000              MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                      ; 2553
 2144  003575' 431150  000020
 2145
 2146  003576' 255400  001777              MWORD   <CONT,A=10,B=0,SOQ,XNOR,D=0,SKCN,MGC=1777>        ; 2554
 2147  003577' 270100  240340
 2148  003600' 255540  470000              MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                      ; 2555
 2149  003601' 431150  000020
 2150  003602' 255600  001777              MWORD   <CONT,A=11,B=1,SOB,XOR,D=1,SKCN,MGC=1777>         ; 2556
 2151  003603' 361110  640340
 2152  003604' 255740  470000              MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                      ; 2557
 2153  003605' 431150  000020
 2154  003606' 256000  001777              MWORD   <CONT,A=12,B=2,SOQ,NAND,D=2,SKCN,MGC=1777>        ; 2560
 2155  003607' 252121  240340
 2156  003610' 256140  470000              MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                      ; 2561
 2157  003611' 431150  000020
 2158  003612' 256200  001777              MWORD   <CONT,A=13,B=3,SOB,AND,D=3,SKCN,MGC=1777>         ; 2562
 2159  003613' 343131  640340
 2160  003614' 256340  470000              MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                      ; 2563
 2161  003615' 431150  000020
 2162  003616' 256400  001777              MWORD   <CONT,A=14,B=4,SOQ,OR,D=4,SKCN,MGC=1777>          ; 2564
 2163  003617' 234142  240340
 2164  003620' 256540  470000              MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                      ; 2565
 2165  003621' 431150  000020
 2166  003622' 256600  001777              MWORD   <CONT,A=15,B=5,SOB,RMIN,D=5,SKCN,MGC=1777>        ; 2566
 2167  003623' 325152  640340
 2168  003624' 256740  470000              MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                      ; 2567
 2169  003625' 431150  000020
 2170  003626' 257000  001777              MWORD   <CONT,A=16,B=6,SOQ,SMIN,D=6,SKCN,MGC=1777>        ; 2570
 2171  003627' 216163  240340
 2172  003630' 257140  470000              MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                      ; 2571
 2173  003631' 431150  000020
 2174  003632' 257200  001777              MWORD   <CONT,A=17,B=7,SOB,PLUS,D=7,SKCN,MGC=1777>        ; 2572
 2175  003633' 307173  640340
 2176  003634' 257340  470000              MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                      ; 2573
 2177  003635' 431150  000020
 2178
 2179  003636' 257400  001777              MWORD   <CONT,A=10,B=0,SOA,PLUS,D=0,SKCN,MGC=1777>        ; 2574
 2180  003637' 400100  240340
 2181  003640' 257540  470000              MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                      ; 2575
 2182  003641' 431150  000020
 2183  003642' 257600  001777              MWORD   <CONT,A=11,B=1,SDA,SMIN,D=1,SKCN,MGC=1777>        ; 2576
 2184  003643' 511110  640340
 2185  003644' 257740  470000              MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                      ; 2577
 2186  003645' 431150  000020
 2187  003646' 260000  001777              MWORD   <CONT,A=12,B=2,SOA,RMIN,D=2,SKCN,MGC=1777>        ; 2600
 2188  003647' 422121  240340
 2189  003650' 260140  470000              MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                      ; 2601
```

```
2190  003651' 431150  000020
2191  003652' 260200  001777                   MWORD   <CONT,A=13,B=3,SDA,OR,D=3,SKCN,MGC=1777>    ; 2602
2192  003653' 533131  640340
2193  003654' 260340  470000                   MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2603
2194  003655' 431150  000020
2195  003656' 260400  001777                   MWORD   <CONT,A=14,B=4,SOA,AND,D=4,SKCN,MGC=1777>   ; 2604
2196  003657' 444142  240340
2197  003660' 260540  470000                   MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2605
2198  003661' 431150  000020
2199  003662' 260600  001777                   MWORD   <CONT,A=15,B=5,SDA,NAND,D=5,SKCN,MGC=1777>  ; 2606
2200  003663' 555152  640340
2201  003664' 260740  470000                   MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2607
2202  003665' 431150  000020
2203  003666' 261000  001777                   MWORD   <CONT,A=16,B=6,SOA,XOR,D=6,SKCN,MGC=1777>   ; 2610
2204  003667' 466163  240340
2205  003670' 261140  470000                   MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2611
2206  003671' 431150  000020
2207  003672' 261200  001777                   MWORD   <CONT,A=17,B=7,SDA,XNOR,D=7,SKCN,MGC=1777>  ; 2612
2208  003673' 577173  640340
2209  003674' 261340  470000                   MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2613
2210  003675' 431150  000020
2211
2212  003676' 261400  001777                   MWORD   <CONT,A=10,B=0,SDQ,XNOR,D=0,SKCN,MGC=1777>  ; 2614
2213  003677' 670100  240340
2214  003700' 261540  470000                   MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2615
2215  003701' 431150  000020
2216  003702' 261600  001777                   MWORD   <CONT,A=11,B=1,SDO,XOR,D=1,SKCN,MGC=1777>   ; 2616
2217  003703' 761110  640340
2218  003704' 261740  470000                   MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2617
2219  003705' 431150  000020
2220  003706' 262000  001777                   MWORD   <CONT,A=12,B=2,SDQ,NAND,D=2,SKCN,MGC=1777>  ; 2620
2221  003707' 652121  240340
2222  003710' 262140  470000                   MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2621
2223  003711' 431150  000020
2224  003712' 262200  001777                   MWORD   <CONT,A=13,B=3,SDO,AND,D=3,SKCN,MGC=1777>   ; 2622
2225  003713' 743131  640340
2226  003714' 262340  470000                   MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2623
2227  003715' 431150  000020
2228  003716' 262400  001777                   MWORD   <CONT,A=14,B=4,SDQ,OR,D=4,SKCN,MGC=1777>    ; 2624
2229  003717' 634142  240340
2230  003720' 262540  470000                   MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2625
2231  003721' 431150  000020
2232  003722' 262600  001777                   MWORD   <CONT,A=15,B=5,SDO,RMIN,D=5,SKCN,MGC=1777>  ; 2626
2233  003723' 725152  640340
2234  003724' 262740  470000                   MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2627
2235  003725' 431150  000020
2236  003726' 263000  001777                   MWORD   <CONT,A=16,B=6,SDQ,SMIN,D=6,SKCN,MGC=1777>  ; 2630
2237  003727' 616163  240340
2238  003730' 263140  470000                   MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2631
2239  003731' 431150  000020
2240  003732' 263200  001777                   MWORD   <CONT,A=17,B=7,SDO,PLUS,D=7,SKCN,MGC=1777>  ; 2632
2241  003733' 707173  640340
2242  003734' 263340  470000                   MWORD   <CJS,J=4047,A=15,SOA,OR,D=1>                ; 2633
2243  003735' 431150  000020
2244
```

```
2245                                   ;       Now do register 16
2246
2247  003736' 263400  020000                   MWORD   <CJS,J=2>                                      ; 2634
2248  003737' 000000  000020
2249  003740' 263500  000000                   MWORD   <CONT,A=0,B=16,SOA,AND,D=2>                    ; 2635
2250  003741' 442007  000340
2251
2252  003742' 263600  001777                   MWORD   <CONT,A=10,B=0,SAQ,PLUS,D=0,SKCN,MGC=1777>     ; 2636
2253  003743' 000100  240340
2254  003744' 263740  520000                   MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                   ; 2637
2255  003745' 431160  000020
2256  003746' 264000  001777                   MWORD   <CONT,A=11,B=1,SAB,SMIN,D=1,SKCN,MGC=1777>     ; 2640
2257  003747' 111110  640340
2258  003750' 264140  520000                   MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                   ; 2641
2259  003751' 431160  000020
2260  003752' 264200  001777                   MWORD   <CONT,A=12,B=2,SAQ,RMIN,D=2,SKCN,MGC=1777>     ; 2642
2261  003753' 022121  240340
2262  003754' 264340  520000                   MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                   ; 2643
2263  003755' 431160  000020
2264  003756' 264400  001777                   MWORD   <CONT,A=13,B=3,SAB,OR,D=3,SKCN,MGC=1777>       ; 2644
2265  003757' 133131  640340
2266  003760' 264540  520000                   MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                   ; 2645
2267  003761' 431160  000020
2268  003762' 264600  001777                   MWORD   <CONT,A=14,B=4,SAQ,AND,D=4,SKCN,MGC=1777>      ; 2646
2269  003763' 044142  240340
2270  003764' 264740  520000                   MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                   ; 2647
2271  003765' 431160  000020
2272  003766' 265000  001777                   MWORD   <CONT,A=15,B=5,SAB,NAND,D=5,SKCN,MGC=1777>     ; 2650
2273  003767' 155152  640340
2274  003770' 265140  520000                   MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                   ; 2651
2275  003771' 431160  000020
2276  003772' 265200  001777                   MWORD   <CONT,A=16,B=6,SAQ,XOR,D=6,SKCN,MGC=1777>      ; 2652
2277  003773' 066163  240340
2278  003774' 265340  520000                   MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                   ; 2653
2279  003775' 431160  000020
2280  003776' 265400  001777                   MWORD   <CONT,A=17,B=7,SAB,XNOR,D=7,SKCN,MGC=1777>     ; 2654
2281  003777' 177173  640340
2282  004000' 265540  520000                   MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                   ; 2655
2283  004001' 431160  000020
2284
2285  004002' 265600  001777                   MWORD   <CONT,A=10,B=0,SOQ,XNOR,D=0,SKCN,MGC=1777>     ; 2656
2286  004003' 270100  240340
2287  004004' 265740  520000                   MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                   ; 2657
2288  004005' 431160  000020
2289  004006' 266000  001777                   MWORD   <CONT,A=11,B=1,SOB,XOR,D=1,SKCN,MGC=1777>      ; 2660
2290  004007' 361110  640340
2291  004010' 266140  520000                   MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                   ; 2661
2292  004011' 431160  000020
2293  004012' 266200  001777                   MWORD   <CONT,A=12,B=2,SOQ,NAND,D=2,SKCN,MGC=1777>     ; 2662
2294  004013' 252121  240340
2295  004014' 266340  520000                   MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                   ; 2663
2296  004015' 431160  000020
2297  004016' 266400  001777                   MWORD   <CONT,A=13,B=3,SOB,AND,D=3,SKCN,MGC=1777>      ; 2664
2298  004017' 343131  640340
2299  004020' 266540  520000                   MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                   ; 2665
```

```
004021' 431160  000020
004022' 266600  001777          MWORD   <CONT,A=14,B=4,SOQ,OR,D=4,SKCN,MGC=1777>      ; 2666
004023' 234142  240340
004024' 266740  520000          MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2667
004025' 431160  000020
004026' 267000  001777          MWORD   <CONT,A=15,B=5,SOB,RMIN,D=5,SKCN,MGC=1777>    ; 2670
004027' 325152  640340
004030' 267140  520000          MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2671
004031' 431160  000020
004032' 267200  001777          MWORD   <CONT,A=16,B=6,SOQ,SMIN,D=6,SKCN,MGC=1777>    ; 2672
004033' 216163  240340
004034' 267340  520000          MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2673
004035' 431160  000020
004036' 267400  001777          MWORD   <CONT,A=17,B=7,SOB,PLUS,D=7,SKCN,MGC=1777>    ; 2674
004037' 307173  640340
004040' 267540  520000          MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2675
004041' 431160  000020

004042' 267600  001777          MWORD   <CONT,A=10,B=0,SOA,PLUS,D=0,SKCN,MGC=1777>    ; 2676
004043' 400100  240340
004044' 267740  520000          MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2677
004045' 431160  000020
004046' 270000  001777          MWORD   <CONT,A=11,B=1,SDA,SMIN,D=1,SKCN,MGC=1777>    ; 2700
004047' 511110  640340
004050' 270140  520000          MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2701
004051' 431160  000020
004052' 270200  001777          MWORD   <CONT,A=12,B=2,SOA,RMIN,D=2,SKCN,MGC=1777>    ; 2702
004053' 422121  240340
004054' 270340  520000          MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2703
004055' 431160  000020
004056' 270400  001777          MWORD   <CONT,A=13,B=3,SDA,OR,D=3,SKCN,MGC=1777>      ; 2704
004057' 533131  640340
004060' 270540  520000          MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2705
004061' 431160  000020
004062' 270600  001777          MWORD   <CONT,A=14,B=4,SOA,AND,D=4,SKCN,MGC=1777>     ; 2706
004063' 444142  240340
004064' 270740  520000          MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2707
004065' 431160  000020
004066' 271000  001777          MWORD   <CONT,A=15,B=5,SDA,NAND,D=5,SKCN,MGC=1777>    ; 2710
004067' 555152  640340
004070' 271140  520000          MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2711
004071' 431160  000020
004072' 271200  001777          MWORD   <CONT,A=16,B=6,SOA,XOR,D=6,SKCN,MGC=1777>     ; 2712
004073' 466163  240340
004074' 271340  520000          MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2713
004075' 431160  000020
004076' 271400  001777          MWORD   <CONT,A=17,B=7,SDA,XNOR,D=7,SKCN,MGC=1777>    ; 2714
004077' 577173  640340
004100' 271540  520000          MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2715
004101' 431160  000020

004102' 271600  001777          MWORD   <CONT,A=10,B=0,SDQ,XNOR,D=0,SKCN,MGC=1777>    ; 2716
004103' 670100  240340
004104' 271740  520000          MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2717
004105' 431160  000020
```

```
004106' 272000  001777                MWORD   <CONT,A=11,B=1,SDO,XOR,D=1,SKCN,MGC=1777>     ; 2720
004107' 761110  640340
004110' 272140  520000                MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2721
004111' 431160  000020
004112' 272200  001777                MWORD   <CONT,A=12,B=2,SDQ,NAND,D=2,SKCN,MGC=1777>    ; 2722
004113' 652121  240340
004114' 272340  520000                MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2723
004115' 431160  000020
004116' 272400  001777                MWORD   <CONT,A=13,B=3,SDO,AND,D=3,SKCN,MGC=1777>     ; 2724
004117' 743131  640340
004120' 272540  520000                MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2725
004121' 431160  000020
004122' 272600  001777                MWORD   <CONT,A=14,B=4,SDQ,OR,D=4,SKCN,MGC=1777>      ; 2726
004123' 634142  240340
004124' 272740  520000                MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2727
004125' 431160  000020
004126' 273000  001777                MWORD   <CONT,A=15,B=5,SDO,RMIN,D=5,SKCN,MGC=1777>    ; 2730
004127' 725152  640340
004130' 273140  520000                MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2731
004131' 431160  000020
004132' 273200  001777                MWORD   <CONT,A=16,B=6,SDQ,SMIN,D=6,SKCN,MGC=1777>    ; 2732
004133' 616163  240340
004134' 273340  520000                MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2733
004135' 431160  000020
004136' 273400  001777                MWORD   <CONT,A=17,B=7,SDO,PLUS,D=7,SKCN,MGC=1777>    ; 2734
004137' 707173  640340
004140' 273540  520000                MWORD   <CJS,J=4052,A=16,SOA,OR,D=1>                  ; 2735
004141' 431160  000020

                              ;       Now do register 17

004142' 273600  020000                MWORD   <CJS,J=2>                                     ; 2736
004143' 000000  000020
004144' 273700  000000                MWORD   <CONT,A=0,B=17,SOA,AND,D=2>                   ; 2737
004145' 442007  400340

004146' 274000  001777                MWORD   <CONT,A=10,B=0,SAQ,PLUS,D=0,SKCN,MGC=1777>    ; 2740
004147' 000100  240340
004150' 274140  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                  ; 2741
004151' 431170  000020
004152' 274200  001777                MWORD   <CONT,A=11,B=1,SAB,SMIN,D=1,SKCN,MGC=1777>    ; 2742
004153' 111110  640340
004154' 274340  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                  ; 2743
004155' 431170  000020
004156' 274400  001777                MWORD   <CONT,A=12,B=2,SAQ,RMIN,D=2,SKCN,MGC=1777>    ; 2744
004157' 022121  240340
004160' 274540  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                  ; 2745
004161' 431170  000020
004162' 274600  001777                MWORD   <CONT,A=13,B=3,SAB,OR,D=3,SKCN,MGC=1777>      ; 2746
004163' 133131  640340
004164' 274740  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                  ; 2747
004165' 431170  000020
004166' 275000  001777                MWORD   <CONT,A=14,B=4,SAQ,AND,D=4,SKCN,MGC=1777>     ; 2750
004167' 044142  240340
004170' 275140  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                  ; 2751
```

```
2410  004171' 431170  000020
2411  004172' 275200  001777                MWORD   <CONT,A=15,B=5,SAB,NAND,D=5,SKCN,MGC=1777>  ; 2752
2412  004173' 155152  640340
2413  004174' 275340  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 2753
2414  004175' 431170  000020
2415  004176' 275400  001777                MWORD   <CONT,A=16,B=6,SAQ,XOR,D=6,SKCN,MGC=1777>   ; 2754
2416  004177' 066163  240340
2417  004200' 275540  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 2755
2418  004201' 431170  000020
2419  004202' 275600  001777                MWORD   <CONT,A=17,B=7,SAB,XNOR,D=7,SKCN,MGC=1777>  ; 2756
2420  004203' 177173  640340
2421  004204' 275740  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 2757
2422  004205' 431170  000020
2423
2424  004206' 276000  001777                MWORD   <CONT,A=10,B=0,SOQ,XNOR,D=0,SKCN,MGC=1777>  ; 2760
2425  004207' 270100  240340
2426  004210' 276140  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 2761
2427  004211' 431170  000020
2428  004212' 276200  001777                MWORD   <CONT,A=11,B=1,SOB,XOR,D=1,SKCN,MGC=1777>   ; 2762
2429  004213' 361110  640340
2430  004214' 276340  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 2763
2431  004215' 431170  000020
2432  004216' 276400  001777                MWORD   <CONT,A=12,B=2,SOQ,NAND,D=2,SKCN,MGC=1777>  ; 2764
2433  004217' 252121  240340
2434  004220' 276540  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 2765
2435  004221' 431170  000020
2436  004222' 276600  001777                MWORD   <CONT,A=13,B=3,SOB,AND,D=3,SKCN,MGC=1777>   ; 2766
2437  004223' 343131  640340
2438  004224' 276740  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 2767
2439  004225' 431170  000020
2440  004226' 277000  001777                MWORD   <CONT,A=14,B=4,SOQ,OR,D=4,SKCN,MGC=1777>    ; 2770
2441  004227' 234142  240340
2442  004230' 277140  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 2771
2443  004231' 431170  000020
2444  004232' 277200  001777                MWORD   <CONT,A=15,B=5,SOB,RMIN,D=5,SKCN,MGC=1777>  ; 2772
2445  004233' 325152  640340
2446  004234' 277340  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 2773
2447  004235' 431170  000020
2448  004236' 277400  001777                MWORD   <CONT,A=16,B=6,SOQ,SMIN,D=6,SKCN,MGC=1777>  ; 2774
2449  004237' 216163  240340
2450  004240' 277540  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 2775
2451  004241' 431170  000020
2452  004242' 277600  001777                MWORD   <CONT,A=17,B=7,SOB,PLUS,D=7,SKCN,MGC=1777>  ; 2776
2453  004243' 307173  640340
2454  004244' 277740  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 2777
2455  004245' 431170  000020
2456
2457  004246' 300000  001777                MWORD   <CONT,A=10,B=0,SOA,PLUS,D=0,SKCN,MGC=1777>  ; 3000
2458  004247' 400100  240340
2459  004250' 300140  550000                MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 3001
2460  004251' 431170  000020
2461  004252' 300200  001777                MWORD   <CONT,A=11,B=1,SDA,SMIN,D=1,SKCN,MGC=1777>  ; 3002
2462  004253' 511110  640340
2463  004254' 300340  550000                MWORD   <CJS,J=4055,A=17,SOA,OR.D=1>                ; 3003
2464  004255' 431170  000020
```

```
004256' 300400  001777          MWORD   <CONT,A=12,B=2,SOA,RMIN,D=2,SKCN,MGC=1777>  ; 3004
004257' 422121  240340
004260' 300540  550000          MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 3005
004261' 431170  000020
004262' 300600  001777          MWORD   <CONT,A=13,B=3,SDA,OR,D=3,SKCN,MGC=1777>    ; 3006
004263' 533131  640340
004264' 300740  550000          MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 3007
004265' 431170  000020
004266' 301000  001777          MWORD   <CONT,A=14,B=4,SOA,AND,D=4,SKCN,MGC=1777>   ; 3010
004267' 444142  240340
004270' 301140  550000          MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 3011
004271' 431170  000020
004272' 301200  001777          MWORD   <CONT,A=15,B=5,SDA,NAND,D=5,SKCN,MGC=1777>  ; 3012
004273' 555152  640340
004274' 301340  550000          MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 3013
004275' 431170  000020
004276' 301400  001777          MWORD   <CONT,A=16,B=6,SOA,XOR,D=6,SKCN,MGC=1777>   ; 3014
004277' 466163  240340
004300' 301540  550000          MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 3015
004301' 431170  000020
004302' 301600  001777          MWORD   <CONT,A=17,B=7,SDA,XNOR,D=7,SKCN,MGC=1777>  ; 3016
004303' 577173  640340
004304' 301740  550000          MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 3017
004305' 431170  000020

004306' 302000  001777          MWORD   <CONT,A=10,B=0,SDQ,XNOR,D=0,SKCN,MGC=1777>  ; 3020
004307' 670100  240340
004310' 302140  550000          MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 3021
004311' 431170  000020
004312' 302200  001777          MWORD   <CONT,A=11,B=1,SDO,XOR,D=1,SKCN,MGC=1777>   ; 3022
004313' 761110  640340
004314' 302340  550000          MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 3023
004315' 431170  000020
004316' 302400  001777          MWORD   <CONT,A=12,B=2,SDQ,NAND,D=2,SKCN,MGC=1777>  ; 3024
004317' 652121  240340
004320' 302540  550000          MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 3025
004321' 431170  000020
004322' 302600  001777          MWORD   <CONT,A=13,B=3,SDO,AND,D=3,SKCN,MGC=1777>   ; 3026
004323' 743131  640340
004324' 302740  550000          MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 3027
004325' 431170  000020
004326' 303000  001777          MWORD   <CONT,A=14,B=4,SDQ,OR,D=4,SKCN,MGC=1777>    ; 3030
004327' 634142  240340
004330' 303140  550000          MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 3031
004331' 431170  000020
004332' 303200  001777          MWORD   <CONT,A=15,B=5,SDO,RMIN,D=5,SKCN,MGC=1777>  ; 3032
004333' 725152  640340
004334' 303340  550000          MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 3033
004335' 431170  000020
004336' 303400  001777          MWORD   <CONT,A=16,B=6,SDQ,SMIN,D=6,SKCN,MGC=1777>  ; 3034
004337' 616163  240340
004340' 303540  550000          MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 3035
004341' 431170  000020
004342' 303600  001777          MWORD   <CONT,A=17,B=7,SDO,PLUS.D=7,SKCN,MGC=1777>  ; 3036
004343' 707173  640340
```

```
004344' 303740  550000          MWORD   <CJS,J=4055,A=17,SOA,OR,D=1>                ; 3037
004345' 431170  000020
004346' 304050  000000          MWORD   <JMAP,J=5000>                               ; 3040 done
004347' 000000  000040
004350' 777777  777777          -1


                        ;#**********************************************************************
                        ; End of 2901 Tests (Part 3)
                        ;#**********************************************************************

                                XLIST

NO ERRORS DETECTED

PROGRAM BREAK IS 004362
CPU TIME USED 09:33.088

255P CORE USED
```

```
DFPTA6  MACRO %53B(1242) 08:42  1-Jul-85 Page S-1
DFPTA6  MAC       18-Apr-85 10:59          SYMBOL TABLE

AAPNT            000000   ext  .LRMIN           000000   spd  .RJ              000000   spd
AEXEC            000023'  ext  .LSOA            000000   spd  .RJMAP           000040   spd
ALU      020000  000000   spd  .LSOB            000000   spd  .RJZ             000000   spd
CALL     200000  000000   spd  .LSOQ            000000   spd  .RMGC            000000   spd
E23              000027   spd  .LSAB            000000   spd  .RNAND   050000  000000   spd
EBUS     400000  000000   spd  .LSAQ            000000   spd  .ROR     030000  000000   spd
ERFLG            000015        .LSDO            000000   spd  .RPLUS           000000   spd
GO       260740  000000        .LSDA            000000   spd  .RRMIN   020000  000000   spd
LAST     010000  000000   spd  .LSDQ            000000   spd  .RSOA    400000  000000   spd
MA32             000031'       .LSKCN           000000   spd  .RSOB    300000  000000   spd
MLAST    400000  000000   spd  .LSMIN           000000   spd  .RSOQ    200000  000000   spd
NDMP     000400  000000   spd  .LXNOR           000000   spd  .RSAB    100000  000000   spd
RTN      263740  000000        .LXOR            000000   spd  .RSAQ            000000   spd
SCOPER   027000  000000        .MA              000017   spd  .RSDO    700000  000000   spd
T32M             000032'       .MAND            000007   spd  .RSDA    500000  000000   spd
TA32             000016'       .MB              000017   spd  .RSDQ    600000  000000   spd
TG32             000011'       .MBAD            000001   spd  .RSKCN           240000   spd
TLOAD            000014'  ext  .MCCFZ           000037   spd  .RSMIN   010000  000000   spd
TRACE            000012'  ext  .MCENA           000001   spd  .RXNOR   070000  000000   spd
TSTA32           000000'  ent  .MCJS            000017   spd  .RXOR    060000  000000   spd
TSTA36           00C000   ext  .MCJV            000017   spd
TSTA37           000000   ext  .MCONT           000017   spd
TSTA40           000000   ext  .MCRTN           000017   spd
TSTA41           000000   ext  .MD              000007   spd
TSTSUB           000000   ext  .MJ              007777   spd
TX32             000030'       .MJMAP           000017   spd
TXALL    060000  000000   spd  .MJZ             000017   spd
Z6               000000'       .MMGC            001777   spd
ZALU     000002  000000   spd  .MNAND           000007   spd
$ARG2            005000        .MOR             000007   spd
$B               000052        .MPLUS           000007   spd
$CHR             000052        .MRMIN           000007   spd
$GARG            000001        .MSOA            000007   spd
%ADDR            003041   spd  .MSOB            000007   spd
%ML      304050  000000   spd  .MSOQ            000007   spd
%MR              000040   spd  .MSAB            000007   spd
.LA              000000   spd  .MSAQ            000007   spd
.LADDR   000100  000000   spd  .MSDO            000007   spd
.LAND            000000   spd  .MSDA            000007   spd
.LB              000000   spd  .MSDQ            000007   spd
.LBAD            000000   spd  .MSKCN           000037   spd
.LCCFZ           000000   spd  .MSMIN           000007   spd
.LCENA           000000   spd  .MXNOR           000007   spd
.LCJS            000000   spd  .MXOR            000007   spd
.LCJV            000000   spd  .RA      000010  000000   spd
.LCONT           000000   spd  .RAND    040000  000000   spd
.LCRTN           000000   spd  .RB              400000   spd
.LD              000000   spd  .RBAD            000001   spd
.LJ              010000   spd  .RCCFZ           020000   spd
.LJMAP           000000   spd  .RCENA   000400  000000   spd
.LJZ             000000   spd  .RCJS            000020   spd
.LMGC            000001   spd  .RCJV            000140   spd
.LNAND           000000   spd  .RCONT           000340   spd
.LOR             000000   spd  .RCRTN           000240   spd
.LPLUS           000000   spd  .RD      001000  000000   spd
```

```
AAPNT   18#   99
AEXEC   18#   87
ALU     63
CALL    99
E23     65
EBUS    63
ERFLG   82    88
LAST    99
MA32    89    99#
MLAST   65
NDMP    63
T32M    64    76    103#
TA32    82#   90
TG32    62    74#
TLOAD   18#   77
TRACE   18#   75
TSTA32  10    62#
TSTA36  14#   66
TSTA37  14#   67
TSTA40  14#   68
TSTA41  14#   69
TSTSUB  18#
TX32    91    95#
TXALL   99
Z6      25#   74
ZALU    63
$ARG2   103#  103   105#  105   111#  111   113#  113   115#  115   117#  117   119#  119
        121#  121   123#  123   125#  125   127#  127   129#  129   131#  131   133#  133
        135#  135   137#  137   139#  139   141#  141   143#  143   145#  145   150#  150
        152#  152   154#  154   159#  159   161#  161   163#  163   168#  168   170#  170
        172#  172   177#  177   179#  179   181#  181   186#  186   188#  188   190#  190
        195#  195   197#  197   199#  199   204#  204   206#  206   208#  208   213#  213
        215#  215   217#  217   222#  222   224#  224   226#  226   231#  231   233#  233
        235#  235   240#  240   242#  242   244#  244   249#  249   251#  251   253#  253
        258#  258   260#  260   262#  262   267#  267   269#  269   271#  271   276#  276
        278#  278   280#  280   285#  285   287#  287   289#  289   294#  294   296#  296
        301#  301   303#  303   306#  306   308#  308   310#  310   312#  312   314#  314
        316#  316   318#  318   320#  320   322#  322   324#  324   326#  326   328#  328
        330#  330   332#  332   334#  334   336#  336   339#  339   341#  341   343#  343
        345#  345   347#  347   349#  349   351#  351   353#  353   355#  355   357#  357
        359#  359   361#  361   363#  363   365#  365   367#  367   369#  369   372#  372
        374#  374   376#  376   378#  378   380#  380   382#  382   384#  384   386#  386
        388#  388   390#  390   392#  392   394#  394   396#  396   398#  398   400#  400
        402#  402   405#  405   407#  407   409#  409   411#  411   413#  413   415#  415
        417#  417   419#  419   421#  421   423#  423   425#  425   427#  427   429#  429
        431#  431   433#  433   435#  435   440#  440   442#  442   445#  445   447#  447
        449#  449   451#  451   453#  453   455#  455   457#  457   459#  459   461#  461
        463#  463   465#  465   467#  467   469#  469   471#  471   473#  473   475#  475
        478#  478   480#  480   482#  482   484#  484   486#  486   488#  488   490#  490
        492#  492   494#  494   496#  496   498#  498   500#  500   502#  502   504#  504
        506#  506   508#  508   511#  511   513#  513   515#  515   517#  517   519#  519
        521#  521   523#  523   525#  525   527#  527   529#  529   531#  531   533#  533
        535#  535   537#  537   539#  539   541#  541   544#  544   546#  546   548#  548
```

```
        550#  550   552#  552   554#  554   556#  556   558#  558   560#  560   562#  562
```

SEQ 1222

SEQ 1235

550#	550	552#	552	554#	554	556#	556	558#	558	560#	560	562#	562
564#	564	566#	566	568#	568	570#	570	572#	572	574#	574	579#	579
581#	581	584#	584	586#	586	588#	588	590#	590	592#	592	594#	594
596#	596	598#	598	600#	600	602#	602	604#	604	606#	606	608#	608
610#	610	612#	612	614#	614	617#	617	619#	619	621#	621	623#	623
625#	625	627#	627	629#	629	631#	631	633#	633	635#	635	637#	637
639#	639	641#	641	643#	643	645#	645	647#	647	650#	650	652#	652
654#	654	656#	656	658#	658	660#	660	662#	662	664#	664	666#	666
668#	668	670#	670	672#	672	674#	674	676#	676	678#	678	680#	680
683#	683	685#	685	687#	687	689#	689	691#	691	693#	693	695#	695
697#	697	699#	699	701#	701	703#	703	705#	705	707#	707	709#	709
711#	711	713#	713	718#	718	720#	720	723#	723	725#	725	727#	727
729#	729	731#	731	733#	733	735#	735	737#	737	739#	739	741#	741
743#	743	745#	745	747#	747	749#	749	751#	751	753#	753	756#	756
758#	758	760#	760	762#	762	764#	764	766#	766	768#	768	770#	770
772#	772	774#	774	776#	776	778#	778	780#	780	782#	782	784#	784
786#	786	789#	789	791#	791	793#	793	795#	795	797#	797	799#	799
801#	801	803#	803	805#	805	807#	807	809#	809	811#	811	813#	813
815#	815	817#	817	819#	819	822#	822	824#	824	826#	826	828#	828
830#	830	832#	832	834#	834	836#	836	838#	838	840#	840	842#	842
844#	844	846#	846	848#	848	850#	850	852#	852	857#	857	859#	859
862#	862	864#	864	866#	866	868#	868	870#	870	872#	872	874#	874
876#	876	878#	878	880#	880	882#	882	884#	884	886#	886	888#	888
890#	890	892#	892	895#	895	897#	897	899#	899	901#	901	903#	903
905#	905	907#	907	909#	909	911#	911	913#	913	915#	915	917#	917
919#	919	921#	921	923#	923	925#	925	928#	928	930#	930	932#	932
934#	934	936#	936	938#	938	940#	940	942#	942	944#	944	946#	946
948#	948	950#	950	952#	952	954#	954	956#	956	958#	958	961#	961
963#	963	965#	965	967#	967	969#	969	971#	971	973#	973	975#	975
977#	977	979#	979	981#	981	983#	983	985#	985	987#	987	989#	989
991#	991	996#	996	998#	998	1001#	1001	1003#	1003	1005#	1005	1007#	1007
1009#	1009	1011#	1011	1013#	1013	1015#	1015	1017#	1017	1019#	1019	1021#	1021
1023#	1023	1025#	1025	1027#	1027	1029#	1029	1031#	1031	1034#	1034	1036#	1036
1038#	1038	1040#	1040	1042#	1042	1044#	1044	1046#	1046	1048#	1048	1050#	1050
1052#	1052	1054#	1054	1056#	1056	1058#	1058	1060#	1060	1062#	1062	1064#	1064
1067#	1067	1069#	1069	1071#	1071	1073#	1073	1075#	1075	1077#	1077	1079#	1079
1081#	1081	1083#	1083	1085#	1085	1087#	1087	1089#	1089	1091#	1091	1093#	1093
1095#	1095	1097#	1097	1100#	1100	1102#	1102	1104#	1104	1106#	1106	1108#	1108
1110#	1110	1112#	1112	1114#	1114	1116#	1116	1118#	1118	1120#	1120	1122#	1122
1124#	1124	1126#	1126	1128#	1128	1130#	1130	1135#	1135	1137#	1137	1140#	1140
1142#	1142	1144#	1144	1146#	1146	1148#	1148	1150#	1150	1152#	1152	1154#	1154
1156#	1156	1158#	1158	1160#	1160	1162#	1162	1164#	1164	1166#	1166	1168#	1168
1170#	1170	1173#	1173	1175#	1175	1177#	1177	1179#	1179	1181#	1181	1183#	1183
1185#	1185	1187#	1187	1189#	1189	1191#	1191	1193#	1193	1195#	1195	1197#	1197
1199#	1199	1201#	1201	1203#	1203	1206#	1206	1208#	1208	1210#	1210	1212#	1212
1214#	1214	1216#	1216	1218#	1218	1220#	1220	1222#	1222	1224#	1224	1226#	1226
1228#	1228	1230#	1230	1232#	1232	1234#	1234	1236#	1236	1239#	1239	1241#	1241
1243#	1243	1245#	1245	1247#	1247	1249#	1249	1251#	1251	1253#	1253	1255#	1255
1257#	1257	1259#	1259	1261#	1261	1263#	1263	1265#	1265	1267#	1267	1269#	1269
1274#	1274	1276#	1276	1279#	1279	1281#	1281	1283#	1283	1285#	1285	1287#	1287
1289#	1289	1291#	1291	1293#	1293	1295#	1295	1297#	1297	1299#	1299	1301#	1301
1303#	1303	1305#	1305	1307#	1307	1309#	1309	1312#	1312	1314#	1314	1316#	1316
1318#	1318	1320#	1320	1322#	1322	1324#	1324	1326#	1326	1328#	1328	1330#	1330

```
1223     1332#  1332   1334#  1334   1336#  1336   1338#  1338   1340#  1340   1342#  1342   1345#  1345     SEQ 1236
         1347#  1347   1349#  1349   1351#  1351   1353#  1353   1355#  1355   1357#  1357   1359#  1359
         1361#  1361   1363#  1363   1365#  1365   1367#  1367   1369#  1369   1371#  1371   1373#  1373
         1375#  1375   1378#  1378   1380#  1380   1382#  1382   1384#  1384   1386#  1386   1388#  1388
         1390#  1390   1392#  1392   1394#  1394   1396#  1396   1398#  1398   1400#  1400   1402#  1402
         1404#  1404   1406#  1406   1408#  1408   1413#  1413   1415#  1415   1418#  1418   1420#  1420
         1422#  1422   1424#  1424   1426#  1426   1428#  1428   1430#  1430   1432#  1432   1434#  1434
         1436#  1436   1438#  1438   1440#  1440   1442#  1442   1444#  1444   1446#  1446   1448#  1448
         1451#  1451   1453#  1453   1455#  1455   1457#  1457   1459#  1459   1461#  1461   1463#  1463
         1465#  1465   1467#  1467   1469#  1469   1471#  1471   1473#  1473   1475#  1475   1477#  1477
         1479#  1479   1481#  1481   1484#  1484   1486#  1486   1488#  1488   1490#  1490   1492#  1492
         1494#  1494   1496#  1496   1498#  1498   1500#  1500   1502#  1502   1504#  1504   1506#  1506
         1508#  1508   1510#  1510   1512#  1512   1514#  1514   1517#  1517   1519#  1519   1521#  1521
         1523#  1523   1525#  1525   1527#  1527   1529#  1529   1531#  1531   1533#  1533   1535#  1535
         1537#  1537   1539#  1539   1541#  1541   1543#  1543   1545#  1545   1547#  1547   1552#  1552
         1554#  1554   1557#  1557   1559#  1559   1561#  1561   1563#  1563   1565#  1565   1567#  1567
         1569#  1569   1571#  1571   1573#  1573   1575#  1575   1577#  1577   1579#  1579   1581#  1581
         1583#  1583   1585#  1585   1587#  1587   1590#  1590   1592#  1592   1594#  1594   1596#  1596
         1598#  1598   1600#  1600   1602#  1602   1604#  1604   1606#  1606   1608#  1608   1610#  1610
         1612#  1612   1614#  1614   1616#  1616   1618#  1618   1620#  1620   1623#  1623   1625#  1625
         1627#  1627   1629#  1629   1631#  1631   1633#  1633   1635#  1635   1637#  1637   1639#  1639
         1641#  1641   1643#  1643   1645#  1645   1647#  1647   1649#  1649   1651#  1651   1653#  1653
         1656#  1656   1658#  1658   1660#  1660   1662#  1662   1664#  1664   1666#  1666   1668#  1668
         1670#  1670   1672#  1672   1674#  1674   1676#  1676   1678#  1678   1680#  1680   1682#  1682
         1684#  1684   1686#  1686   1691#  1691   1693#  1693   1696#  1696   1698#  1698   1700#  1700
         1702#  1702   1704#  1704   1706#  1706   1708#  1708   1710#  1710   1712#  1712   1714#  1714
         1716#  1716   1718#  1718   1720#  1720   1722#  1722   1724#  1724   1726#  1726   1729#  1729
         1731#  1731   1733#  1733   1735#  1735   1737#  1737   1739#  1739   1741#  1741   1743#  1743
         1745#  1745   1747#  1747   1749#  1749   1751#  1751   1753#  1753   1755#  1755   1757#  1757
         1759#  1759   1762#  1762   1764#  1764   1766#  1766   1768#  1768   1770#  1770   1772#  1772
         1774#  1774   1776#  1776   1778#  1778   1780#  1780   1782#  1782   1784#  1784   1786#  1786
         1788#  1788   1790#  1790   1792#  1792   1795#  1795   1797#  1797   1799#  1799   1801#  1801
         1803#  1803   1805#  1805   1807#  1807   1809#  1809   1811#  1811   1813#  1813   1815#  1815
         1817#  1817   1819#  1819   1821#  1821   1823#  1823   1825#  1825   1830#  1830   1832#  1832
         1835#  1835   1837#  1837   1839#  1839   1841#  1841   1843#  1843   1845#  1845   1847#  1847
         1849#  1849   1851#  1851   1853#  1853   1855#  1855   1857#  1857   1859#  1859   1861#  1861
         1863#  1863   1865#  1865   1868#  1868   1870#  1870   1872#  1872   1874#  1874   1876#  1876
         1878#  1878   1880#  1880   1882#  1882   1884#  1884   1886#  1886   1888#  1888   1890#  1890
         1892#  1892   1894#  1894   1896#  1896   1898#  1898   1901#  1901   1903#  1903   1905#  1905
         1907#  1907   1909#  1909   1911#  1911   1913#  1913   1915#  1915   1917#  1917   1919#  1919
         1921#  1921   1923#  1923   1925#  1925   1927#  1927   1929#  1929   1931#  1931   1934#  1934
         1936#  1936   1938#  1938   1940#  1940   1942#  1942   1944#  1944   1946#  1946   1948#  1948
         1950#  1950   1952#  1952   1954#  1954   1956#  1956   1958#  1958   1960#  1960   1962#  1962
         1964#  1964   1969#  1969   1971#  1971   1974#  1974   1976#  1976   1978#  1978   1980#  1980
         1982#  1982   1984#  1984   1986#  1986   1988#  1988   1990#  1990   1992#  1992   1994#  1994
         1996#  1996   1998#  1998   2000#  2000   2002#  2002   2004#  2004   2007#  2007   2009#  2009
         2011#  2011   2013#  2013   2015#  2015   2017#  2017   2019#  2019   2021#  2021   2023#  2023
         2025#  2025   2027#  2027   2029#  2029   2031#  2031   2033#  2033   2035#  2035   2037#  2037
         2040#  2040   2042#  2042   2044#  2044   2046#  2046   2048#  2048   2050#  2050   2052#  2052
         2054#  2054   2056#  2056   2058#  2058   2060#  2060   2062#  2062   2064#  2064   2066#  2066
         2068#  2068   2070#  2070   2073#  2073   2075#  2075   2077#  2077   2079#  2079   2081#  2081
         2083#  2083   2085#  2085   2087#  2087   2089#  2089   2091#  2091   2093#  2093   2095#  2095
         2097#  2097   2099#  2099   2101#  2101   2103#  2103   2108#  2108   2110#  2110   2113#  2113
```

```
1224     2115#  2115   2117#  2117   2119#  2119   2121#  2121   2123#  2123   2125#  2125   2127#  2127
         2129#  2129   2131#  2131   2133#  2133   2135#  2135   2137#  2137   2139#  2139   2141#  2141     SEQ 1237
         2143#  2143   2146#  2146   2148#  2148   2150#  2150   2152#  2152   2154#  2154   2156#  2156
```

```
1224    2115#  2115   2117#  2117   2119#  2119   2121#  2121   2123#  2123   2125#  2125   2127#  2127    SEQ 1237
        2129#  2129   2131#  2131   2133#  2133   2135#  2135   2137#  2137   2139#  2139   2141#  2141
        2143#  2143   2146#  2146   2148#  2148   2150#  2150   2152#  2152   2154#  2154   2156#  2156
        2158#  2158   2160#  2160   2162#  2162   2164#  2164   2166#  2166   2168#  2168   2170#  2170
        2172#  2172   2174#  2174   2176#  2176   2179#  2179   2181#  2181   2183#  2183   2185#  2185
        2187#  2187   2189#  2189   2191#  2191   2193#  2193   2195#  2195   2197#  2197   2199#  2199
        2201#  2201   2203#  2203   2205#  2205   2207#  2207   2209#  2209   2212#  2212   2214#  2214
        2216#  2216   2218#  2218   2220#  2220   2222#  2222   2224#  2224   2226#  2226   2228#  2228
        2230#  2230   2232#  2232   2234#  2234   2236#  2236   2238#  2238   2240#  2240   2242#  2242
        2247#  2247   2249#  2249   2252#  2252   2254#  2254   2256#  2256   2258#  2258   2260#  2260
        2262#  2262   2264#  2264   2266#  2266   2268#  2268   2270#  2270   2272#  2272   2274#  2274
        2276#  2276   2278#  2278   2280#  2280   2282#  2282   2285#  2285   2287#  2287   2289#  2289
        2291#  2291   2293#  2293   2295#  2295   2297#  2297   2299#  2299   2301#  2301   2303#  2303
        2305#  2305   2307#  2307   2309#  2309   2311#  2311   2313#  2313   2315#  2315   2318#  2318
        2320#  2320   2322#  2322   2324#  2324   2326#  2326   2328#  2328   2330#  2330   2332#  2332
        2334#  2334   2336#  2336   2338#  2338   2340#  2340   2342#  2342   2344#  2344   2346#  2346
        2348#  2348   2351#  2351   2353#  2353   2355#  2355   2357#  2357   2359#  2359   2361#  2361
        2363#  2363   2365#  2365   2367#  2367   2369#  2369   2371#  2371   2373#  2373   2375#  2375
        2377#  2377   2379#  2379   2381#  2381   2386#  2386   2388#  2388   2391#  2391   2393#  2393
        2395#  2395   2397#  2397   2399#  2399   2401#  2401   2403#  2403   2405#  2405   2407#  2407
        2409#  2409   2411#  2411   2413#  2413   2415#  2415   2417#  2417   2419#  2419   2421#  2421
        2424#  2424   2426#  2426   2428#  2428   2430#  2430   2432#  2432   2434#  2434   2436#  2436
        2438#  2438   2440#  2440   2442#  2442   2444#  2444   2446#  2446   2448#  2448   2450#  2450
        2452#  2452   2454#  2454   2457#  2457   2459#  2459   2461#  2461   2463#  2463   2465#  2465
        2467#  2467   2469#  2469   2471#  2471   2473#  2473   2475#  2475   2477#  2477   2479#  2479
        2481#  2481   2483#  2483   2485#  2485   2487#  2487   2490#  2490   2492#  2492   2494#  2494
        2496#  2496   2498#  2498   2500#  2500   2502#  2502   2504#  2504   2506#  2506   2508#  2508
        2510#  2510   2512#  2512   2514#  2514   2516#  2516   2518#  2518   2520#  2520   2522#  2522
$B       103#   103    105#   105    111#   111    113#   113    115#   115    117#   117    119#   119
         121#   121    123#   123    125#   125    127#   127    129#   129    131#   131    133#   133
         135#   135    137#   137    139#   139    141#   141    143#   143    145#   145    150#   150
         152#   152    154#   154    159#   159    161#   161    163#   163    168#   168    170#   170
         172#   172    177#   177    179#   179    181#   181    186#   186    188#   188    190#   190
         195#   195    197#   197    199#   199    204#   204    206#   206    208#   208    213#   213
         215#   215    217#   217    222#   222    224#   224    226#   226    231#   231    233#   233
         235#   235    240#   240    242#   242    244#   244    249#   249    251#   251    253#   253
         258#   258    260#   260    262#   262    267#   267    269#   269    271#   271    276#   276
         278#   278    280#   280    285#   285    287#   287    289#   289    294#   294    296#   296
         301#   301    303#   303    306#   306    308#   308    310#   310    312#   312    314#   314
         316#   316    318#   318    320#   320    322#   322    324#   324    326#   326    328#   328
         330#   330    332#   332    334#   334    336#   336    339#   339    341#   341    343#   343
         345#   345    347#   347    349#   349    351#   351    353#   353    355#   355    357#   357
         359#   359    361#   361    363#   363    365#   365    367#   367    369#   369    372#   372
         374#   374    376#   376    378#   378    380#   380    382#   382    384#   384    386#   386
         388#   388    390#   390    392#   392    394#   394    396#   396    398#   398    400#   400
         402#   402    405#   405    407#   407    409#   409    411#   411    413#   413    415#   415
         417#   417    419#   419    421#   421    423#   423    425#   425    427#   427    429#   429
         431#   431    433#   433    435#   435    440#   440    442#   442    445#   445    447#   447
         449#   449    451#   451    453#   453    455#   455    457#   457    459#   459    461#   461
         463#   463    465#   465    467#   467    469#   469    471#   471    473#   473    475#   475
         478#   478    480#   480    482#   482    484#   484    486#   486    488#   488    490#   490
         492#   492    494#   494    496#   496    498#   498    500#   500    502#   502    504#   504
         506#   506    508#   508    511#   511    513#   513    515#   515    517#   517    519#   519
```

```
1225     521#   521    523#   523    525#   525    527#   527    529#   529    531#   531    533#   533    SEQ 1238
         535#   535    537#   537    539#   539    541#   541    544#   544    546#   546    548#   548
         550#   550    552#   552    554#   554    556#   556    558#   558    560#   560    562#   562
```

1225

SEQ 1238

521#	521	523#	523	525#	525	527#	527	529#	529	531#	531	533#	533		
535#	535	537#	537	539#	539	541#	541	544#	544	546#	546	548#	548		
550#	550	552#	552	554#	554	556#	556	558#	558	560#	560	562#	562		
564#	564	566#	566	568#	568	570#	570	572#	572	574#	574	579#	579		
581#	581	584#	584	586#	586	588#	588	590#	590	592#	592	594#	594		
596#	596	598#	598	600#	600	602#	602	604#	604	606#	606	608#	608		
610#	610	612#	612	614#	614	617#	617	619#	619	621#	621	623#	623		
625#	625	627#	627	629#	629	631#	631	633#	633	635#	635	637#	637		
639#	639	641#	641	643#	643	645#	645	647#	647	650#	650	652#	652		
654#	654	656#	656	658#	658	660#	660	662#	662	664#	664	666#	666		
668#	668	670#	670	672#	672	674#	674	676#	676	678#	678	680#	680		
683#	683	685#	685	687#	687	689#	689	691#	691	693#	693	695#	695		
697#	697	699#	699	701#	701	703#	703	705#	705	707#	707	709#	709		
711#	711	713#	713	718#	718	720#	720	723#	723	725#	725	727#	727		
729#	729	731#	731	733#	733	735#	735	737#	737	739#	739	741#	741		
743#	743	745#	745	747#	747	749#	749	751#	751	753#	753	756#	756		
758#	758	760#	760	762#	762	764#	764	766#	766	768#	768	770#	770		
772#	772	774#	774	776#	776	778#	778	780#	780	782#	782	784#	784		
786#	786	789#	789	791#	791	793#	793	795#	795	797#	797	799#	799		
801#	801	803#	803	805#	805	807#	807	809#	809	811#	811	813#	813		
815#	815	817#	817	819#	819	822#	822	824#	824	826#	826	828#	828		
830#	830	832#	832	834#	834	836#	836	838#	838	840#	840	842#	842		
844#	844	846#	846	848#	848	850#	850	852#	852	857#	857	859#	859		
862#	862	864#	864	866#	866	868#	868	870#	870	872#	872	874#	874		
876#	876	878#	878	880#	880	882#	882	884#	884	886#	886	888#	888		
890#	890	892#	892	895#	895	897#	897	899#	899	901#	901	903#	903		
905#	905	907#	907	909#	909	911#	911	913#	913	915#	915	917#	917		
919#	919	921#	921	923#	923	925#	925	928#	928	930#	930	932#	932		
934#	934	936#	936	938#	938	940#	940	942#	942	944#	944	946#	946		
948#	948	950#	950	952#	952	954#	954	956#	956	958#	958	961#	961		
963#	963	965#	965	967#	967	969#	969	971#	971	973#	973	975#	975		
977#	977	979#	979	981#	981	983#	983	985#	985	987#	987	989#	989		
991#	991	996#	996	998#	998	1001#	1001	1003#	1003	1005#	1005	1007#	1007		
1009#	1009	1011#	1011	1013#	1013	1015#	1015	1017#	1017	1019#	1019	1021#	1021		
1023#	1023	1025#	1025	1027#	1027	1029#	1029	1031#	1031	1034#	1034	1036#	1036		
1038#	1038	1040#	1040	1042#	1042	1044#	1044	1046#	1046	1048#	1048	1050#	1050		
1052#	1052	1054#	1054	1056#	1056	1058#	1058	1060#	1060	1062#	1062	1064#	1064		
1067#	1067	1069#	1069	1071#	1071	1073#	1073	1075#	1075	1077#	1077	1079#	1079		
1081#	1081	1083#	1083	1085#	1085	1087#	1087	1089#	1089	1091#	1091	1093#	1093		
1095#	1095	1097#	1097	1100#	1100	1102#	1102	1104#	1104	1106#	1106	1108#	1108		
1110#	1110	1112#	1112	1114#	1114	1116#	1116	1118#	1118	1120#	1120	1122#	1122		
1124#	1124	1126#	1126	1128#	1128	1130#	1130	1135#	1135	1137#	1137	1140#	1140		
1142#	1142	1144#	1144	1146#	1146	1148#	1148	1150#	1150	1152#	1152	1154#	1154		
1156#	1156	1158#	1158	1160#	1160	1162#	1162	1164#	1164	1166#	1166	1168#	1168		
1170#	1170	1173#	1173	1175#	1175	1177#	1177	1179#	1179	1181#	1181	1183#	1183		
1185#	1185	1187#	1187	1189#	1189	1191#	1191	1193#	1193	1195#	1195	1197#	1197		
1199#	1199	1201#	1201	1203#	1203	1206#	1206	1208#	1208	1210#	1210	1212#	1212		
1214#	1214	1216#	1216	1218#	1218	1220#	1220	1222#	1222	1224#	1224	1226#	1226		
1228#	1228	1230#	1230	1232#	1232	1234#	1234	1236#	1236	1239#	1239	1241#	1241		
1243#	1243	1245#	1245	1247#	1247	1249#	1249	1251#	1251	1253#	1253	1255#	1255		
1257#	1257	1259#	1259	1261#	1261	1263#	1263	1265#	1265	1267#	1267	1269#	1269		
1274#	1274	1276#	1276	1279#	1279	1281#	1281	1283#	1283	1285#	1285	1287#	1287		
1289#	1289	1291#	1291	1293#	1293	1295#	1295	1297#	1297	1299#	1299	1301#	1301		

1226

SEQ 1239

1303#	1303	1305#	1305	1307#	1307	1309#	1309	1312#	1312	1314#	1314	1316#	1316
1318#	1318	1320#	1320	1322#	1322	1324#	1324	1326#	1326	1328#	1328	1330#	1330
1332#	1332	1334#	1334	1336#	1336	1338#	1338	1340#	1340	1342#	1342	1345#	1345

1226

SEQ 1239

1303#	1303	1305#	1305	1307#	1307	1309#	1309	1312#	1312	1314#	1314	1316#	1316
1318#	1318	1320#	1320	1322#	1322	1324#	1324	1326#	1326	1328#	1328	1330#	1330
1332#	1332	1334#	1334	1336#	1336	1338#	1338	1340#	1340	1342#	1342	1345#	1345
1347#	1347	1349#	1349	1351#	1351	1353#	1353	1355#	1355	1357#	1357	1359#	1359
1361#	1361	1363#	1363	1365#	1365	1367#	1367	1369#	1369	1371#	1371	1373#	1373
1375#	1375	1378#	1378	1380#	1380	1382#	1382	1384#	1384	1386#	1386	1388#	1388
1390#	1390	1392#	1392	1394#	1394	1396#	1396	1398#	1398	1400#	1400	1402#	1402
1404#	1404	1406#	1406	1408#	1408	1413#	1413	1415#	1415	1418#	1418	1420#	1420
1422#	1422	1424#	1424	1426#	1426	1428#	1428	1430#	1430	1432#	1432	1434#	1434
1436#	1436	1438#	1438	1440#	1440	1442#	1442	1444#	1444	1446#	1446	1448#	1448
1451#	1451	1453#	1453	1455#	1455	1457#	1457	1459#	1459	1461#	1461	1463#	1463
1465#	1465	1467#	1467	1469#	1469	1471#	1471	1473#	1473	1475#	1475	1477#	1477
1479#	1479	1481#	1481	1484#	1484	1486#	1486	1488#	1488	1490#	1490	1492#	1492
1494#	1494	1496#	1496	1498#	1498	1500#	1500	1502#	1502	1504#	1504	1506#	1506
1508#	1508	1510#	1510	1512#	1512	1514#	1514	1517#	1517	1519#	1519	1521#	1521
1523#	1523	1525#	1525	1527#	1527	1529#	1529	1531#	1531	1533#	1533	1535#	1535
1537#	1537	1539#	1539	1541#	1541	1543#	1543	1545#	1545	1547#	1547	1552#	1552
1554#	1554	1557#	1557	1559#	1559	1561#	1561	1563#	1563	1565#	1565	1567#	1567
1569#	1569	1571#	1571	1573#	1573	1575#	1575	1577#	1577	1579#	1579	1581#	1581
1583#	1583	1585#	1585	1587#	1587	1590#	1590	1592#	1592	1594#	1594	1596#	1596
1598#	1598	1600#	1600	1602#	1602	1604#	1604	1606#	1606	1608#	1608	1610#	1610
1612#	1612	1614#	1614	1616#	1616	1618#	1618	1620#	1620	1623#	1623	1625#	1625
1627#	1627	1629#	1629	1631#	1631	1633#	1633	1635#	1635	1637#	1637	1639#	1639
1641#	1641	1643#	1643	1645#	1645	1647#	1647	1649#	1649	1651#	1651	1653#	1653
1656#	1656	1658#	1658	1660#	1660	1662#	1662	1664#	1664	1666#	1666	1668#	1668
1670#	1670	1672#	1672	1674#	1674	1676#	1676	1678#	1678	1680#	1680	1682#	1682
1684#	1684	1686#	1686	1691#	1691	1693#	1693	1696#	1696	1698#	1698	1700#	1700
1702#	1702	1704#	1704	1706#	1706	1708#	1708	1710#	1710	1712#	1712	1714#	1714
1716#	1716	1718#	1718	1720#	1720	1722#	1722	1724#	1724	1726#	1726	1729#	1729
1731#	1731	1733#	1733	1735#	1735	1737#	1737	1739#	1739	1741#	1741	1743#	1743
1745#	1745	1747#	1747	1749#	1749	1751#	1751	1753#	1753	1755#	1755	1757#	1757
1759#	1759	1762#	1762	1764#	1764	1766#	1766	1768#	1768	1770#	1770	1772#	1772
1774#	1774	1776#	1776	1778#	1778	1780#	1780	1782#	1782	1784#	1784	1786#	1786
1788#	1788	1790#	1790	1792#	1792	1795#	1795	1797#	1797	1799#	1799	1801#	1801
1803#	1803	1805#	1805	1807#	1807	1809#	1809	1811#	1811	1813#	1813	1815#	1815
1817#	1817	1819#	1819	1821#	1821	1823#	1823	1825#	1825	1830#	1830	1832#	1832
1835#	1835	1837#	1837	1839#	1839	1841#	1841	1843#	1843	1845#	1845	1847#	1847
1849#	1849	1851#	1851	1853#	1853	1855#	1855	1857#	1857	1859#	1859	1861#	1861
1863#	1863	1865#	1865	1868#	1868	1870#	1870	1872#	1872	1874#	1874	1876#	1876
1878#	1878	1880#	1880	1882#	1882	1884#	1884	1886#	1886	1888#	1888	1890#	1890
1892#	1892	1894#	1894	1896#	1896	1898#	1898	1901#	1901	1903#	1903	1905#	1905
1907#	1907	1909#	1909	1911#	1911	1913#	1913	1915#	1915	1917#	1917	1919#	1919
1921#	1921	1923#	1923	1925#	1925	1927#	1927	1929#	1929	1931#	1931	1934#	1934
1936#	1936	1938#	1938	1940#	1940	1942#	1942	1944#	1944	1946#	1946	1948#	1948
1950#	1950	1952#	1952	1954#	1954	1956#	1956	1958#	1958	1960#	1960	1962#	1962
1964#	1964	1969#	1969	1971#	1971	1974#	1974	1976#	1976	1978#	1978	1980#	1980
1982#	1982	1984#	1984	1986#	1986	1988#	1988	1990#	1990	1992#	1992	1994#	1994
1996#	1996	1998#	1998	2000#	2000	2002#	2002	2004#	2004	2007#	2007	2009#	2009
2011#	2011	2013#	2013	2015#	2015	2017#	2017	2019#	2019	2021#	2021	2023#	2023
2025#	2025	2027#	2027	2029#	2029	2031#	2031	2033#	2033	2035#	2035	2037#	2037
2040#	2040	2042#	2042	2044#	2044	2046#	2046	2048#	2048	2050#	2050	2052#	2052
2054#	2054	2056#	2056	2058#	2058	2060#	2060	2062#	2062	2064#	2064	2066#	2066
2068#	2068	2070#	2070	2073#	2073	2075#	2075	2077#	2077	2079#	2079	2081#	2081

Q 1227

SEQ 1240

2083#	2083	2085#	2085	2087#	2087	2089#	2089	2091#	2091	2093#	2093	2095#	2095
2097#	2097	2099#	2099	2101#	2101	2103#	2103	2108#	2108	2110#	2110	2113#	2113

```
Q 1227  2083#  2083    2085#  2085    2087#  2087    2089#  2089    2091#  2091    2093#  2093    2095#  2095    SEQ 1240
        2097#  2097    2099#  2099    2101#  2101    2103#  2103    2108#  2108    2110#  2110    2113#  2113
        2115#  2115    2117#  2117    2119#  2119    2121#  2121    2123#  2123    2125#  2125    2127#  2127
        2129#  2129    2131#  2131    2133#  2133    2135#  2135    2137#  2137    2139#  2139    2141#  2141
        2143#  2143    2146#  2146    2148#  2148    2150#  2150    2152#  2152    2154#  2154    2156#  2156
        2158#  2158    2160#  2160    2162#  2162    2164#  2164    2166#  2166    2168#  2168    2170#  2170
        2172#  2172    2174#  2174    2176#  2176    2179#  2179    2181#  2181    2183#  2183    2185#  2185
        2187#  2187    2189#  2189    2191#  2191    2193#  2193    2195#  2195    2197#  2197    2199#  2199
        2201#  2201    2203#  2203    2205#  2205    2207#  2207    2209#  2209    2212#  2212    2214#  2214
        2216#  2216    2218#  2218    2220#  2220    2222#  2222    2224#  2224    2226#  2226    2228#  2228
        2230#  2230    2232#  2232    2234#  2234    2236#  2236    2238#  2238    2240#  2240    2242#  2242
        2247#  2247    2249#  2249    2252#  2252    2254#  2254    2256#  2256    2258#  2258    2260#  2260
        2262#  2262    2264#  2264    2266#  2266    2268#  2268    2270#  2270    2272#  2272    2274#  2274
        2276#  2276    2278#  2278    2280#  2280    2282#  2282    2285#  2285    2287#  2287    2289#  2289
        2291#  2291    2293#  2293    2295#  2295    2297#  2297    2299#  2299    2301#  2301    2303#  2303
        2305#  2305    2307#  2307    2309#  2309    2311#  2311    2313#  2313    2315#  2315    2318#  2318
        2320#  2320    2322#  2322    2324#  2324    2326#  2326    2328#  2328    2330#  2330    2332#  2332
        2334#  2334    2336#  2336    2338#  2338    2340#  2340    2342#  2342    2344#  2344    2346#  2346
        2348#  2348    2351#  2351    2353#  2353    2355#  2355    2357#  2357    2359#  2359    2361#  2361
        2363#  2363    2365#  2365    2367#  2367    2369#  2369    2371#  2371    2373#  2373    2375#  2375
        2377#  2377    2379#  2379    2381#  2381    2386#  2386    2388#  2388    2391#  2391    2393#  2393
        2395#  2395    2397#  2397    2399#  2399    2401#  2401    2403#  2403    2405#  2405    2407#  2407
        2409#  2409    2411#  2411    2413#  2413    2415#  2415    2417#  2417    2419#  2419    2421#  2421
        2424#  2424    2426#  2426    2428#  2428    2430#  2430    2432#  2432    2434#  2434    2436#  2436
        2438#  2438    2440#  2440    2442#  2442    2444#  2444    2446#  2446    2448#  2448    2450#  2450
        2452#  2452    2454#  2454    2457#  2457    2459#  2459    2461#  2461    2463#  2463    2465#  2465
        2467#  2467    2469#  2469    2471#  2471    2473#  2473    2475#  2475    2477#  2477    2479#  2479
        2481#  2481    2483#  2483    2485#  2485    2487#  2487    2490#  2490    2492#  2492    2494#  2494
        2496#  2496    2498#  2498    2500#  2500    2502#  2502    2504#  2504    2506#  2506    2508#  2508
        2510#  2510    2512#  2512    2514#  2514    2516#  2516    2518#  2518    2520#  2520    2522#  2522
$CHR     103#   103     105#   105     111#   111     113#   113     115#   115     117#   117     119#   119
         121#   121     123#   123     125#   125     127#   127     129#   129     131#   131     133#   133
         135#   135     137#   137     139#   139     141#   141     143#   143     145#   145     150#   150
         152#   152     154#   154     159#   159     161#   161     163#   163     168#   168     170#   170
         172#   172     177#   177     179#   179     181#   181     186#   186     188#   188     190#   190
         195#   195     197#   197     199#   199     204#   204     206#   206     208#   208     213#   213
         215#   215     217#   217     222#   222     224#   224     226#   226     231#   231     233#   233
         235#   235     240#   240     242#   242     244#   244     249#   249     251#   251     253#   253
         258#   258     260#   260     262#   262     267#   267     269#   269     271#   271     276#   276
         278#   278     280#   280     285#   285     287#   287     289#   289     294#   294     296#   296
         301#   301     303#   303     306#   306     308#   308     310#   310     312#   312     314#   314
         316#   316     318#   318     320#   320     322#   322     324#   324     326#   326     328#   328
         330#   330     332#   332     334#   334     336#   336     339#   339     341#   341     343#   343
         345#   345     347#   347     349#   349     351#   351     353#   353     355#   355     357#   357
         359#   359     361#   361     363#   363     365#   365     367#   367     369#   369     372#   372
         374#   374     376#   376     378#   378     380#   380     382#   382     384#   384     386#   386
         388#   388     390#   390     392#   392     394#   394     396#   396     398#   398     400#   400
         402#   402     405#   405     407#   407     409#   409     411#   411     413#   413     415#   415
         417#   417     419#   419     421#   421     423#   423     425#   425     427#   427     429#   429
         431#   431     433#   433     435#   435     440#   440     442#   442     445#   445     447#   447
         449#   449     451#   451     453#   453     455#   455     457#   457     459#   459     461#   461
         463#   463     465#   465     467#   467     469#   469     471#   471     473#   473     475#   475
         478#   478     480#   480     482#   482     484#   484     486#   486     488#   488     490#   490
```

```
EQ 1228  492#   492     494#   494     496#   496     498#   498     500#   500     502#   502     504#   504    SEQ 1241
         506#   506     508#   508     511#   511     513#   513     515#   515     517#   517     519#   519
```

```
Q 1228

 492#   492   494#   494   496#   496   498#   498   500#   500   502#   502   504#   504     SEQ 1241
 506#   506   508#   508   511#   511   513#   513   515#   515   517#   517   519#   519
 521#   521   523#   523   525#   525   527#   527   529#   529   531#   531   533#   533
 535#   535   537#   537   539#   539   541#   541   544#   544   546#   546   548#   548
 550#   550   552#   552   554#   554   556#   556   558#   558   560#   560   562#   562
 564#   564   566#   566   568#   568   570#   570   572#   572   574#   574   579#   579
 581#   581   584#   584   586#   586   588#   588   590#   590   592#   592   594#   594
 596#   596   598#   598   600#   600   602#   602   604#   604   606#   606   608#   608
 610#   610   612#   612   614#   614   617#   617   619#   619   621#   621   623#   623
 625#   625   627#   627   629#   629   631#   631   633#   633   635#   635   637#   637
 639#   639   641#   641   643#   643   645#   645   647#   647   650#   650   652#   652
 654#   654   656#   656   658#   658   660#   660   662#   662   664#   664   666#   666
 668#   668   670#   670   672#   672   674#   674   676#   676   678#   678   680#   680
 683#   683   685#   685   687#   687   689#   689   691#   691   693#   693   695#   695
 697#   697   699#   699   701#   701   703#   703   705#   705   707#   707   709#   709
 711#   711   713#   713   718#   718   720#   720   723#   723   725#   725   727#   727
 729#   729   731#   731   733#   733   735#   735   737#   737   739#   739   741#   741
 743#   743   745#   745   747#   747   749#   749   751#   751   753#   753   756#   756
 758#   758   760#   760   762#   762   764#   764   766#   766   768#   768   770#   770
 772#   772   774#   774   776#   776   778#   778   780#   780   782#   782   784#   784
 786#   786   789#   789   791#   791   793#   793   795#   795   797#   797   799#   799
 801#   801   803#   803   805#   805   807#   807   809#   809   811#   811   813#   813
 815#   815   817#   817   819#   819   822#   822   824#   824   826#   826   828#   828
 830#   830   832#   832   834#   834   836#   836   838#   838   840#   840   842#   842
 844#   844   846#   846   848#   848   850#   850   852#   852   857#   857   859#   859
 862#   862   864#   864   866#   866   868#   868   870#   870   872#   872   874#   874
 876#   876   878#   878   880#   880   882#   882   884#   884   886#   886   888#   888
 890#   890   892#   892   895#   895   897#   897   899#   899   901#   901   903#   903
 905#   905   907#   907   909#   909   911#   911   913#   913   915#   915   917#   917
 919#   919   921#   921   923#   923   925#   925   928#   928   930#   930   932#   932
 934#   934   936#   936   938#   938   940#   940   942#   942   944#   944   946#   946
 948#   948   950#   950   952#   952   954#   954   956#   956   958#   958   961#   961
 963#   963   965#   965   967#   967   969#   969   971#   971   973#   973   975#   975
 977#   977   979#   979   981#   981   983#   983   985#   985   987#   987   989#   989
 991#   991   996#   996   998#   998  1001#  1001  1003#  1003  1005#  1005  1007#  1007
1009#  1009  1011#  1011  1013#  1013  1015#  1015  1017#  1017  1019#  1019  1021#  1021
1023#  1023  1025#  1025  1027#  1027  1029#  1029  1031#  1031  1034#  1034  1036#  1036
1038#  1038  1040#  1040  1042#  1042  1044#  1044  1046#  1046  1048#  1048  1050#  1050
1052#  1052  1054#  1054  1056#  1056  1058#  1058  1060#  1060  1062#  1062  1064#  1064
1067#  1067  1069#  1069  1071#  1071  1073#  1073  1075#  1075  1077#  1077  1079#  1079
1081#  1081  1083#  1083  1085#  1085  1087#  1087  1089#  1089  1091#  1091  1093#  1093
1095#  1095  1097#  1097  1100#  1100  1102#  1102  1104#  1104  1106#  1106  1108#  1108
1110#  1110  1112#  1112  1114#  1114  1116#  1116  1118#  1118  1120#  1120  1122#  1122
1124#  1124  1126#  1126  1128#  1128  1130#  1130  1135#  1135  1137#  1137  1140#  1140
1142#  1142  1144#  1144  1146#  1146  1148#  1148  1150#   150  1152#  1152  1154#  1154
1156#  1156  1158#  1158  1160#  1160  1162#  1162  1164#   164  1166#  1166  1168#  1168
1170#  1170  1173#  1173  1175#  1175  1177#  1177  1179#    79  1181#  1181  1183#  1183
1185#  1185  1187#  1187  1189#  1189  1191#  1191  119   1193  1195#  1195  1197#  1197
1199#  1199  1201#  1201  1203#  1203  1206#  1206  1208#  1208  1210#  1210  1212#  1212
1214#  1214  1216#  1216  1218#  1218  1220#  1220  1222#  1222  1224#  1224  1226#  1226
1228#  1228  1230#  1230  1232#  1232  1234#  1234  1236#  1236  1239#  1239  1241#  1241
1243#  1243  1245#  1245  1247#  1247  1249#  1249  1251#  1251  1253#  1253  1255#  1255
1257#  1257  1259#  1259  1261#  1261  1263#  1263  1265#  1265  1267#  1267  1269#  1269

EQ 1229

1274#  1274  1276#  1276  1279#  1279  1281#  1281  1283#  1283  1285#  1285  1287#  1287     SEQ 1242
1289#  1289  1291#  1291  1293#  1293  1295#  1295  1297#  1297  1299#  1299  1301#  1301
```

EQ 1229

```
1274#  1274  1276#  1276  1279#  1279  1281#  1281  1283#  1283  1285#  1285  1287#  1287
1289#  1289  1291#  1291  1293#  1293  1295#  1295  1297#  1297  1299#  1299  1301#  1301
1303#  1303  1305#  1305  1307#  1307  1309#  1309  1312#  1312  1314#  1314  1316#  1316
1318#  1318  1320#  1320  1322#  1322  1324#  1324  1326#  1326  1328#  1328  1330#  1330
1332#  1332  1334#  1334  1336#  1336  1338#  1338  1340#  1340  1342#  1342  1345#  1345
1347#  1347  1349#  1349  1351#  1351  1353#  1353  1355#  1355  1357#  1357  1359#  1359
1361#  1361  1363#  1363  1365#  1365  1367#  1367  1369#  1369  1371#  1371  1373#  1373
1375#  1375  1378#  1378  1380#  1380  1382#  1382  1384#  1384  1386#  1386  1388#  1388
1390#  1390  1392#  1392  1394#  1394  1396#  1396  1398#  1398  1400#  1400  1402#  1402
1404#  1404  1406#  1406  1408#  1408  1413#  1413  1415#  1415  1418#  1418  1420#  1420
1422#  1422  1424#  1424  1426#  1426  1428#  1428  1430#  1430  1432#  1432  1434#  1434
1436#  1436  1438#  1438  1440#  1440  1442#  1442  1444#  1444  1446#  1446  1448#  1448
1451#  1451  1453#  1453  1455#  1455  1457#  1457  1459#  1459  1461#  1461  1463#  1463
1465#  1465  1467#  1467  1469#  1469  1471#  1471  1473#  1473  1475#  1475  1477#  1477
1479#  1479  1481#  1481  1484#  1484  1486#  1486  1488#  1488  1490#  1490  1492#  1492
1494#  1494  1496#  1496  1498#  1498  1500#  1500  1502#  1502  1504#  1504  1506#  1506
1508#  1508  1510#  1510  1512#  1512  1514#  1514  1517#  1517  1519#  1519  1521#  1521
1523#  1523  1525#  1525  1527#  1527  1529#  1529  1531#  1531  1533#  1533  1535#  1535
1537#  1537  1539#  1539  1541#  1541  1543#  1543  1545#  1545  1547#  1547  1552#  1552
1554#  1554  1557#  1557  1559#  1559  1561#  1561  1563#  1563  1565#  1565  1567#  1567
1569#  1569  1571#  1571  1573#  1573  1575#  1575  1577#  1577  1579#  1579  1581#  1581
1583#  1583  1585#  1585  1587#  1587  1590#  1590  1592#  1592  1594#  1594  1596#  1596
1598#  1598  1600#  1600  1602#  1602  1604#  1604  1606#  1606  1608#  1608  1610#  1610
1612#  1612  1614#  1614  1616#  1616  1618#  1618  1620#  1620  1623#  1623  1625#  1625
1627#  1627  1629#  1629  1631#  1631  1633#  1633  1635#  1635  1637#  1637  1639#  1639
1641#  1641  1643#  1643  1645#  1645  1647#  1647  1649#  1649  1651#  1651  1653#  1653
1656#  1656  1658#  1658  1660#  1660  1662#  1662  1664#  1664  1666#  1666  1668#  1668
1670#  1670  1672#  1672  1674#  1674  1676#  1676  1678#  1678  1680#  1680  1682#  1682
1684#  1684  1686#  1686  1691#  1691  1693#  1693  1696#  1696  1698#  1698  1700#  1700
1702#  1702  1704#  1704  1706#  1706  1708#  1708  1710#  1710  1712#  1712  1714#  1714
1716#  1716  1718#  1718  1720#  1720  1722#  1722  1724#  1724  1726#  1726  1729#  1729
1731#  1731  1733#  1733  1735#  1735  1737#  1737  1739#  1739  1741#  1741  1743#  1743
1745#  1745  1747#  1747  1749#  1749  1751#  1751  1753#  1753  1755#  1755  1757#  1757
1759#  1759  1762#  1762  1764#  1764  1766#  1766  1768#  1768  1770#  1770  1772#  1772
1774#  1774  1776#  1776  1778#  1778  1780#  1780  1782#  1782  1784#  1784  1786#  1786
1788#  1788  1790#  1790  1792#  1792  1795#  1795  1797#  1797  1799#  1799  1801#  1801
1803#  1803  1805#  1805  1807#  1807  1809#  1809  1811#  1811  1813#  1813  1815#  1815
1817#  1817  1819#  1819  1821#  1821  1823#  1823  1825#  1825  1830#  1830  1832#  1832
1835#  1835  1837#  1837  1839#  1839  1841#  1841  1843#  1843  1845#  1845  1847#  1847
1849#  1849  1851#  1851  1853#  1853  1855#  1855  1857#  1857  1859#  1859  1861#  1861
1863#  1863  1865#  1865  1868#  1868  1870#  1870  1872#  1872  1874#  1874  1876#  1876
1878#  1878  1880#  1880  1882#  1882  1884#  1884  1886#  1886  1888#  1888  1890#  1890
1892#  1892  1894#  1894  1896#  1896  1898#  1898  1901#  1901  1903#  1903  1905#  1905
1907#  1907  1909#  1909  1911#  1911  1913#  1913  1915#  1915  1917#  1917  1919#  1919
1921#  1921  1923#  1923  1925#  1925  1927#  1927  1929#  1929  1931#  1931  1934#  1934
1936#  1936  1938#  1938  1940#  1940  1942#  1942  1944#  1944  1946#  1946  1948#  1948
1950#  1950  1952#  1952  1954#  1954  1956#  1956  1958#  1958  1960#  1960  1962#  1962
1964#  1964  1969#  1969  1971#  1971  1974#  1974  1976#  1976  1978#  1978  1980#  1980
1982#  1982  1984#  1984  1986#  1986  1988#  1988  1990#  1990  1992#  1992  1994#  1994
1996#  1996  1998#  1998  2000#  2000  2002#  2002  2004#  2004  2007#  2007  2009#  2009
2011#  2011  2013#  2013  2015#  2015  2017#  2017  2019#  2019  2021#  2021  2023#  2023
2025#  2025  2027#  2027  2029#  2029  2031#  2031  2033#  2033  2035#  2035  2037#  2037
2040#  2040  2042#  2042  2044#  2044  2046#  2046  2048#  2048  2050#  2050  2052#  2052
```

SEQ 1242

EQ 1230

```
2054#  2054  2056#  2056  2058#  2058  2060#  2060  2062#  2062  2064#  2064  2066#  2066
2068#  2068  2070#  2070  2073#  2073  2075#  2075  2077#  2077  2079#  2079  2081#  2081
```

SEQ 1243

```
SEQ 1230          2054#  2054   2056#  2056   2058#  2058   2060#  2060   2062#  2062   2064#  2064   2066#  2066    SEQ 1243
                  2068#  2068   2070#  2070   2073#  2073   2075#  2075   2077#  2077   2079#  2079   2081#  2081
                  2083#  2083   2085#  2085   2087#  2087   2089#  2089   2091#  2091   2093#  2093   2095#  2095
                  2097#  2097   2099#  2099   2101#  2101   2103#  2103   2108#  2108   2110#  2110   2113#  2113
                  2115#  2115   2117#  2117   2119#  2119   2121#  2121   2123#  2123   2125#  2125   2127#  2127
                  2129#  2129   2131#  2131   2133#  2133   2135#  2135   2137#  2137   2139#  2139   2141#  2141
                  2143#  2143   2146#  2146   2148#  2148   2150#  2150   2152#  2152   2154#  2154   2156#  2156
                  2158#  2158   2160#  2160   2162#  2162   2164#  2164   2166#  2166   2168#  2168   2170#  2170
                  2172#  2172   2174#  2174   2176#  2176   2179#  2179   2181#  2181   2183#  2183   2185#  2185
                  2187#  2187   2189#  2189   2191#  2191   2193#  2193   2195#  2195   2197#  2197   2199#  2199
                  2201#  2201   2203#  2203   2205#  2205   2207#  2207   2209#  2209   2212#  2212   2214#  2214
                  2216#  2216   2218#  2218   2220#  2220   2222#  2222   2224#  2224   2226#  2226   2228#  2228
                  2230#  2230   2232#  2232   2234#  2234   2236#  2236   2238#  2238   2240#  2240   2242#  2242
                  2247#  2247   2249#  2249   2252#  2252   2254#  2254   2256#  2256   2258#  2258   2260#  2260
                  2262#  2262   2264#  2264   2266#  2266   2268#  2268   2270#  2270   2272#  2272   2274#  2274
                  2276#  2276   2278#  2278   2280#  2280   2282#  2282   2285#  2285   2287#  2287   2289#  2289
                  2291#  2291   2293#  2293   2295#  2295   2297#  2297   2299#  2299   2301#  2301   2303#  2303
                  2305#  2305   2307#  2307   2309#  2309   2311#  2311   2313#  2313   2315#  2315   2318#  2318
                  2320#  2320   2322#  2322   2324#  2324   2326#  2326   2328#  2328   2330#  2330   2332#  2332
                  2334#  2334   2336#  2336   2338#  2338   2340#  2340   2342#  2342   2344#  2344   2346#  2346
                  2348#  2348   2351#  2351   2353#  2353   2355#  2355   2357#  2357   2359#  2359   2361#  2361
                  2363#  2363   2365#  2365   2367#  2367   2369#  2369   2371#  2371   2373#  2373   2375#  2375
                  2377#  2377   2379#  2379   2381#  2381   2386#  2386   2388#  2388   2391#  2391   2393#  2393
                  2395#  2395   2397#  2397   2399#  2399   2401#  2401   2403#  2403   2405#  2405   2407#  2407
                  2409#  2409   2411#  2411   2413#  2413   2415#  2415   2417#  2417   2419#  2419   2421#  2421
                  2424#  2424   2426#  2426   2428#  2428   2430#  2430   2432#  2432   2434#  2434   2436#  2436
                  2438#  2438   2440#  2440   2442#  2442   2444#  2444   2446#  2446   2448#  2448   2450#  2450
                  2452#  2452   2454#  2454   2457#  2457   2459#  2459   2461#  2461   2463#  2463   2465#  2465
                  2467#  2467   2469#  2469   2471#  2471   2473#  2473   2475#  2475   2477#  2477   2479#  2479
                  2481#  2481   2483#  2483   2485#  2485   2487#  2487   2490#  2490   2492#  2492   2494#  2494
                  2496#  2496   2498#  2498   2500#  2500   2502#  2502   2504#  2504   2506#  2506   2508#  2508
                  2510#  2510   2512#  2512   2514#  2514   2516#  2516   2518#  2518   2520#  2520   2522#  2522
        $GARG      103#   103    105#   105    111#   111    113#   113    115#   115    117#   117    119#   119
                   121#   121    123#   123    125#   125    127#   127    129#   129    131#   131    133#   133
                   135#   135    137#   137    139#   139    141#   141    143#   143    145#   145    150#   150
                   152#   152    154#   154    159#   159    161#   161    163#   163    168#   168    170#   170
                   172#   172    177#   177    179#   179    181#   181    186#   186    188#   188    190#   190
                   195#   195    197#   197    199#   199    204#   204    206#   206    208#   208    213#   213
                   215#   215    217#   217    222#   222    224#   224    226#   226    231#   231    233#   233
                   235#   235    240#   240    242#   242    244#   244    249#   249    251#   251    253#   253
                   258#   258    260#   260    262#   262    267#   267    269#   269    271#   271    276#   276
                   278#   278    280#   280    285#   285    287#   287    289#   289    294#   294    296#   296
                   301#   301    303#   303    306#   306    308#   308    310#   310    312#   312    314#   314
                   316#   316    318#   318    320#   320    322#   322    324#   324    326#   326    328#   328
                   330#   330    332#   332    334#   334    336#   336    339#   339    341#   341    343#   343
                   345#   345    347#   347    349#   349    351#   351    353#   353    355#   355    357#   357
                   359#   359    361#   361    363#   363    365#   365    367#   367    369#   369    372#   372
                   374#   374    376#   376    378#   378    380#   380    382#   382    384#   384    386#   386
                   388#   388    390#   390    392#   392    394#   394    396#   396    398#   398    400#   400
                   402#   402    405#   405    407#   407    409#   409    411#   411    413#   413    415#   415
                   417#   417    419#   419    421#   421    423#   423    425#   425    427#   427    429#   429
                   431#   431    433#   433    435#   435    440#   440    442#   442    445#   445    447#   447
                   449#   449    451#   451    453#   453    455#   455    457#   457    459#   459    461#   461
```

```
SEQ 1231           463#   463    465#   465    467#   467    469#   469    471#   471    473#   473    475#   475    SEQ 1244
                   478#   478    480#   480    482#   482    484#   484    486#   486    488#   488    490#   490
```

```
SEQ 1231   463#  463   465#  465   467#  467   469#  469   471#  471   473#  473   475#  475   SEQ 1244
           478#  478   480#  480   482#  482   484#  484   486#  486   488#  488   490#  490
           492#  492   494#  494   496#  496   498#  498   500#  500   502#  502   504#  504
           506#  506   508#  508   511#  511   513#  513   515#  515   517#  517   519#  519
           521#  521   523#  523   525#  525   527#  527   529#  529   531#  531   533#  533
           535#  535   537#  537   539#  539   541#  541   544#  544   546#  546   548#  548
           550#  550   552#  552   554#  554   556#  556   558#  558   560#  560   562#  562
           564#  564   566#  566   568#  568   570#  570   572#  572   574#  574   579#  579
           581#  581   584#  584   586#  586   588#  588   590#  590   592#  592   594#  594
           596#  596   598#  598   600#  600   602#  602   604#  604   606#  606   608#  608
           610#  610   612#  612   614#  614   617#  617   619#  619   621#  621   623#  623
           625#  625   627#  627   629#  629   631#  631   633#  633   635#  635   637#  637
           639#  639   641#  641   643#  643   645#  645   647#  647   650#  650   652#  652
           654#  654   656#  656   658#  658   660#  660   662#  662   664#  664   666#  666
           668#  668   670#  670   672#  672   674#  674   676#  676   678#  678   680#  680
           683#  683   685#  685   687#  687   689#  689   691#  691   693#  693   695#  695
           697#  697   699#  699   701#  701   703#  703   705#  705   707#  707   709#  709
           711#  711   713#  713   718#  718   720#  720   723#  723   725#  725   727#  727
           729#  729   731#  731   733#  733   735#  735   737#  737   739#  739   741#  741
           743#  743   745#  745   747#  747   749#  749   751#  751   753#  753   756#  756
           758#  758   760#  760   762#  762   764#  764   766#  766   768#  768   770#  770
           772#  772   774#  774   776#  776   778#  778   780#  780   782#  782   784#  784
           786#  786   789#  789   791#  791   793#  793   795#  795   797#  797   799#  799
           801#  801   803#  803   805#  805   807#  807   809#  809   811#  811   813#  813
           815#  815   817#  817   819#  819   822#  822   824#  824   826#  826   828#  828
           830#  830   832#  832   834#  834   836#  836   838#  838   840#  840   842#  842
           844#  844   846#  846   848#  848   850#  850   852#  852   857#  857   859#  859
           862#  862   864#  864   866#  866   868#  868   870#  870   872#  872   874#  874
           876#  876   878#  878   880#  880   882#  882   884#  884   886#  886   888#  888
           890#  890   892#  892   895#  895   897#  897   899#  899   901#  901   903#  903
           905#  905   907#  907   909#  909   911#  911   913#  913   915#  915   917#  917
           919#  919   921#  921   923#  923   925#  925   928#  928   930#  930   932#  932
           934#  934   936#  936   938#  938   940#  940   942#  942   944#  944   946#  946
           948#  948   950#  950   952#  952   954#  954   956#  956   958#  958   961#  961
           963#  963   965#  965   967#  967   969#  969   971#  971   973#  973   975#  975
           977#  977   979#  979   981#  981   983#  983   985#  985   987#  987   989#  989
           991#  991   996#  996   998#  998  1001# 1001  1003# 1003  1005# 1005  1007# 1007
          1009# 1009  1011# 1011  1013# 1013  1015# 1015  1017# 1017  1019# 1019  1021# 1021
          1023# 1023  1025# 1025  1027# 1027  1029# 1029  1031# 1031  1034# 1034  1036# 1036
          1038# 1038  1040# 1040  1042# 1042  1044# 1044  1046# 1046  1048# 1048  1050# 1050
          1052# 1052  1054# 1054  1056# 1056  1058# 1058  1060# 1060  1062# 1062  1064# 1064
          1067# 1067  1069# 1069  1071# 1071  1073# 1073  1075# 1075  1077# 1077  1079# 1079
          1081# 1081  1083# 1083  1085# 1085  1087# 1087  1089# 1089  1091# 1091  1093# 1093
          1095# 1095  1097# 1097  1100# 1100  1102# 1102  1104# 1104  1106# 1106  1108# 1108
          1110# 1110  1112# 1112  1114# 1114  1116# 1116  1118# 1118  1120# 1120  1122# 1122
          1124# 1124  1126# 1126  1128# 1128  1130# 1130  1135# 1135  1137# 1137  1140# 1140
          1142# 1142  1144# 1144  1146# 1146  1148# 1148  1150# 1150  1152# 1152  1154# 1154
          1156# 1156  1158# 1158  1160# 1160  1162# 1162  1164# 1164  1166# 1166  1168# 1168
          1170# 1170  1173# 1173  1175# 1175  1177# 1177  1179# 1179  1181# 1181  1183# 1183
          1185# 1185  1187# 1187  1189# 1189  1191# 1191  1193# 1193  1195# 1195  1197# 1197
          1199# 1199  1201# 1201  1203# 1203  1206# 1206  1208# 1208  1210# 1210  1212# 1212
          1214# 1214  1216# 1216  1218# 1218  1220# 1220  1222# 1222  1224# 1224  1226# 1226
          1228# 1228  1230# 1230  1232# 1232  1234# 1234  1236# 1236  1239# 1239  1241# 1241

SEQ 1232  1243# 1243  1245# 1245  1247# 1247  1249# 1249  1251# 1251  1253# 1253  1255# 1255  SEQ 1245
SEQ 1232   1243#  1243   1245#  1245   1247#  1247   1249#  1249   1251#  1251   1253#  1253   1255#  1255   SEQ 1245
           1257#  1257   1259#  1259   1261#  1261   1263#  1263   1265#  1265   1267#  1267   1269#  1269
           1274#  1274   1276#  1276   1279#  1279   1281#  1281   1283#  1283   1285#  1285   1287#  1287
           1289#  1289   1291#  1291   1293#  1293   1295#  1295   1297#  1297   1299#  1299   1301#  1301
           1303#  1303   1305#  1305   1307#  1307   1309#  1309   1312#  1312   1314#  1314   1316#  1316
           1318#  1318   1320#  1320   1322#  1322   1324#  1324   1326#  1326   1328#  1328   1330#  1330
           1332#  1332   1334#  1334   1336#  1336   1338#  1338   1340#  1340   1342#  1342   1345#  1345
           1347#  1347   1349#  1349   1351#  1351   1353#  1353   1355#  1355   1357#  1357   1359#  1359
           1361#  1361   1363#  1363   1365#  1365   1367#  1367   1369#  1369   1371#  1371   1373#  1373
           1375#  1375   1378#  1378   1380#  1380   1382#  1382   1384#  1384   1386#  1386   1388#  1388
           1390#  1390   1392#  1392   1394#  1394   1396#  1396   1398#  1398   1400#  1400   1402#  1402
           1404#  1404   1406#  1406   1408#  1408   1413#  1413   1415#  1415   1418#  1418   1420#  1420
           1422#  1422   1424#  1424   1426#  1426   1428#  1428   1430#  1430   1432#  1432   1434#  1434
           1436#  1436   1438#  1438   1440#  1440   1442#  1442   1444#  1444   1446#  1446   1448#  1448
           1451#  1451   1453#  1453   1455#  1455   1457#  1457   1459#  1459   1461#  1461   1463#  1463
           1465#  1465   1467#  1467   1469#  1469   1471#  1471   1473#  1473   1475#  1475   1477#  1477
           1479#  1479   1481#  1481   1484#  1484   1486#  1486   1488#  1488   1490#  1490   1492#  1492
           1494#  1494   1496#  1496   1498#  1498   1500#  1500   1502#  1502   1504#  1504   1506#  1506
           1508#  1508   1510#  1510   1512#  1512   1514#  1514   1517#  1517   1519#  1519   1521#  1521
           1523#  1523   1525#  1525   1527#  1527   1529#  1529   1531#  1531   1533#  1533   1535#  1535
           1537#  1537   1539#  1539   1541#  1541   1543#  1543   1545#  1545   1547#  1547   1552#  1552
           1554#  1554   1557#  1557   1559#  1559   1561#  1561   1563#  1563   1565#  1565   1567#  1567
           1569#  1569   1571#  1571   1573#  1573   1575#  1575   1577#  1577   1579#  1579   1581#  1581
           1583#  1583   1585#  1585   1587#  1587   1590#  1590   1592#  1592   1594#  1594   1596#  1596
           1598#  1598   1600#  1600   1602#  1602   1604#  1604   1606#  1606   1608#  1608   1610#  1610
           1612#  1612   1614#  1614   1616#  1616   1618#  1618   1620#  1620   1623#  1623   1625#  1625
           1627#  1627   1629#  1629   1631#  1631   1633#  1633   1635#  1635   1637#  1637   1639#  1639
           1641#  1641   1643#  1643   1645#  1645   1647#  1647   1649#  1649   1651#  1651   1653#  1653
           1656#  1656   1658#  1658   1660#  1660   1662#  1662   1664#  1664   1666#  1666   1668#  1668
           1670#  1670   1672#  1672   1674#  1674   1676#  1676   1678#  1678   1680#  1680   1682#  1682
           1684#  1684   1686#  1686   1691#  1691   1693#  1693   1696#  1696   1698#  1698   1700#  1700
           1702#  1702   1704#  1704   1706#  1706   1708#  1708   1710#  1710   1712#  1712   1714#  1714
           1716#  1716   1718#  1718   1720#  1720   1722#  1722   1724#  1724   1726#  1726   1729#  1729
           1731#  1731   1733#  1733   1735#  1735   1737#  1737   1739#  1739   1741#  1741   1743#  1743
           1745#  1745   1747#  1747   1749#  1749   1751#  1751   1753#  1753   1755#  1755   1757#  1757
           1759#  1759   1762#  1762   1764#  1764   1766#  1766   1768#  1768   1770#  1770   1772#  1772
           1774#  1774   1776#  1776   1778#  1778   1780#  1780   1782#  1782   1784#  1784   1786#  1786
           1788#  1788   1790#  1790   1792#  1792   1795#  1795   1797#  1797   1799#  1799   1801#  1801
           1803#  1803   1805#  1805   1807#  1807   1809#  1809   1811#  1811   1813#  1813   1815#  1815
           1817#  1817   1819#  1819   1821#  1821   1823#  1823   1825#  1825   1830#  1830   1832#  1832
           1835#  1835   1837#  1837   1839#  1839   1841#  1841   1843#  1843   1845#  1845   1847#  1847
           1849#  1849   1851#  1851   1853#  1853   1855#  1855   1857#  1857   1859#  1859   1861#  1861
           1863#  1863   1865#  1865   1868#  1868   1870#  1870   1872#  1872   1874#  1874   1876#  1876
           1878#  1878   1880#  1880   1882#  1882   1884#  1884   1886#  1886   1888#  1888   1890#  1890
           1892#  1892   1894#  1894   1896#  1896   1898#  1898   1901#  1901   1903#  1903   1905#  1905
           1907#  1907   1909#  1909   1911#  1911   1913#  1913   1915#  1915   1917#  1917   1919#  1919
           1921#  1921   1923#  1923   1925#  1925   1927#  1927   1929#  1929   1931#  1931   1934#  1934
           1936#  1936   1938#  1938   1940#  1940   1942#  1942   1944#  1944   1946#  1946   1948#  1948
           1950#  1950   1952#  1952   1954#  1954   1956#  1956   1958#  1958   1960#  1960   1962#  1962
           1964#  1964   1969#  1969   1971#  1971   1974#  1974   1976#  1976   1978#  1978   1980#  1980
           1982#  1982   1984#  1984   1986#  1986   1988#  1988   1990#  1990   1992#  1992   1994#  1994
           1996#  1996   1998#  1998   2000#  2000   2002#  2002   2004#  2004   2007#  2007   2009#  2009
           2011#  2011   2013#  2013   2015#  2015   2017#  2017   2019#  2019   2021#  2021   2023#  2023
```

```
SEQ 1233   2025#  2025   2027#  2027   2029#  2029   2031#  2031   2033#  2033   2035#  2035   2037#  2037   SEQ 1246
SEQ 1233  2025#   2025    2027#   2027    2029#   2029    2031#   2031    2033#   2033    2035#   2035    2037#   2037      SEQ 1246
          2040#   2040    2042#   2042    2044#   2044    2046#   2046    2048#   2048    2050#   2050    2052#   2052
          2054#   2054    2056#   2056    2058#   2058    2060#   2060    2062#   2062    2064#   2064    2066#   2066
          2068#   2068    2070#   2070    2073#   2073    2075#   2075    2077#   2077    2079#   2079    2081#   2081
          2083#   2083    2085#   2085    2087#   2087    2089#   2089    2091#   2091    2093#   2093    2095#   2095
          2097#   2097    2099#   2099    2101#   2101    2103#   2103    2108#   2108    2110#   2110    2113#   2113
          2115#   2115    2117#   2117    2119#   2119    2121#   2121    2123#   2123    2125#   2125    2127#   2127
          2129#   2129    2131#   2131    2133#   2133    2135#   2135    2137#   2137    2139#   2139    2141#   2141
          2143#   2143    2146#   2146    2148#   2148    2150#   2150    2152#   2152    2154#   2154    2156#   2156
          2158#   2158    2160#   2160    2162#   2162    2164#   2164    2166#   2166    2168#   2168    2170#   2170
          2172#   2172    2174#   2174    2176#   2176    2179#   2179    2181#   2181    2183#   2183    2185#   2185
          2187#   2187    2189#   2189    2191#   2191    2193#   2193    2195#   2195    2197#   2197    2199#   2199
          2201#   2201    2203#   2203    2205#   2205    2207#   2207    2209#   2209    2212#   2212    2214#   2214
          2216#   2216    2218#   2218    2220#   2220    2222#   2222    2224#   2224    2226#   2226    2228#   2228
          2230#   2230    2232#   2232    2234#   2234    2236#   2236    2238#   2238    2240#   2240    2242#   2242
          2247#   2247    2249#   2249    2252#   2252    2254#   2254    2256#   2256    2258#   2258    2260#   2260
          2262#   2262    2264#   2264    2266#   2266    2268#   2268    2270#   2270    2272#   2272    2274#   2274
          2276#   2276    2278#   2278    2280#   2280    2282#   2282    2285#   2285    2287#   2287    2289#   2289
          2291#   2291    2293#   2293    2295#   2295    2297#   2297    2299#   2299    2301#   2301    2303#   2303
          2305#   2305    2307#   2307    2309#   2309    2311#   2311    2313#   2313    2315#   2315    2318#   2318
          2320#   2320    2322#   2322    2324#   2324    2326#   2326    2328#   2328    2330#   2330    2332#   2332
          2334#   2334    2336#   2336    2338#   2338    2340#   2340    2342#   2342    2344#   2344    2346#   2346
          2348#   2348    2351#   2351    2353#   2353    2355#   2355    2357#   2357    2359#   2359    2361#   2361
          2363#   2363    2365#   2365    2367#   2367    2369#   2369    2371#   2371    2373#   2373    2375#   2375
          2377#   2377    2379#   2379    2381#   2381    2386#   2386    2388#   2388    2391#   2391    2393#   2393
          2395#   2395    2397#   2397    2399#   2399    2401#   2401    2403#   2403    2405#   2405    2407#   2407
          2409#   2409    2411#   2411    2413#   2413    2415#   2415    2417#   2417    2419#   2419    2421#   2421
          2424#   2424    2426#   2426    2428#   2428    2430#   2430    2432#   2432    2434#   2434    2436#   2436
          2438#   2438    2440#   2440    2442#   2442    2444#   2444    2446#   2446    2448#   2448    2450#   2450
          2452#   2452    2454#   2454    2457#   2457    2459#   2459    2461#   2461    2463#   2463    2465#   2465
          2467#   2467    2469#   2469    2471#   2471    2473#   2473    2475#   2475    2477#   2477    2479#   2479
          2481#   2481    2483#   2483    2485#   2485    2487#   2487    2490#   2490    2492#   2492    2494#   2494
          2496#   2496    2498#   2498    2500#   2500    2502#   2502    2504#   2504    2506#   2506    2508#   2508
          2510#   2510    2512#   2512    2514#   2514    2516#   2516    2518#   2518    2520#   2520    2522#   2522
%ADDR      103#    103     105     105#    111     111#    113     113#    115     115#    117     117#    119     119#
           121     121#    123     123#    125     125#    127     127#    129     129#    131     131#    133     133#
           135     135#    137     137#    139     139#    141     141#    143     143#    145     145#    150#    150
           152     152#    154     154#    159     159#    161     161#    163     163#    168     168#    170     170#
           172     172#    177     177#    179     179#    181     181#    186     186#    188     188#    190     190#
           195     195#    197     197#    199     199#    204     204#    206     206#    208     208#    213     213#
           215     215#    217     217#    222     222#    224     224#    226     226#    231     231#    233     233#
           235     235#    240     240#    242     242#    244     244#    249     249#    251     251#    253     253#
           258     258#    260     260#    262     262#    267     267#    269     269#    271     271#    276     276#
           278     278#    280     280#    285     285#    287     287#    289     289#    294#    294     296#    296
           301#    301     303     303#    306     306#    308     308#    310     310#    312     312#    314     314#
           316     316#    318     318#    320     320#    322     322#    324     324#    326     326#    328     328#
           330     330#    332     332#    334     334#    336     336#    339     339#    341     341#    343     343#
           345     345#    347     347#    349     349#    351     351#    353     353#    355     355#    357     357#
           359     359#    361     361#    363     363#    365     365#    367     367#    369     369#    372     372#
           374     374#    376     376#    378     378#    380     380#    382     382#    384     384#    386     386#
           388     388#    390     390#    392     392#    394     394#    396     396#    398     398#    400     400#
           402     402#    405     405#    407     407#    409     409#    411     411#    413     413#    415     415#
           417     417#    419     419#    421     421#    423     423#    425     425#    427     427#    429     429#
```

```
           431     431#    433     433#    435     435#    440     440#    442     442#    445     445#    447     447#     SEQ 1247
```

SEQ 1234

431	431#	433	433#	435	435#	440	440#	442	442#	445	445#	447	447#
449	449#	451	451#	453	453#	455	455#	457	457#	459	459#	461	461#
463	463#	465	465#	467	467#	469	469#	471	471#	473	473#	475	475#
478	478#	480	480#	482	482#	484	484#	486	486#	488	488#	490	490#
492	492#	494	494#	496	496#	498	498#	500	500#	502	502#	504	504#
506	506#	508	508#	511	511#	513	513#	515	515#	517	517#	519	519#
521	521#	523	523#	525	525#	527	527#	529	529#	531	531#	533	533#
535	535#	537	537#	539	539#	541	541#	544	544#	546	546#	548	548#
550	550#	552	552#	554	554#	556	556#	558	558#	560	560#	562	562#
564	564#	566	566#	568	568#	570	570#	572	572#	574	574#	579	579#
581	581#	584	584#	586	586#	588	588#	590	590#	592	592#	594	594#
596	596#	598	598#	600	600#	602	602#	604	604#	606	606#	608	608#
610	610#	612	612#	614	614#	617	617#	619	619#	621	621#	623	623#
625	625#	627	627#	629	629#	631	631#	633	633#	635	635#	637	637#
639	639#	641	641#	643	643#	645	645#	647	647#	650	650#	652	652#
654	654#	656	656#	658	658#	660	660#	662	662#	664	664#	666	666#
668	668#	670	670#	672	672#	674	674#	676	676#	678	678#	680	680#
683	683#	685	685#	687	687#	689	689#	691	691#	693	693#	695	695#
697	697#	699	699#	701	701#	703	703#	705	705#	707	707#	709	709#
711	711#	713	713#	718	718#	720	720#	723	723#	725	725#	727	727#
729	729#	731	731#	733	733#	735	735#	737	737#	739	739#	741	741#
743	743#	745	745#	747	747#	749	749#	751	751#	753	753#	756	756#
758	758#	760	760#	762	762#	764	764#	766	766#	768	768#	770	770#
772	772#	774	774#	776	776#	778	778#	780	780#	782	782#	784	784#
786	786#	789	789#	791	791#	793	793#	795	795#	797	797#	799	799#
801	801#	803	803#	805	805#	807	807#	809	809#	811	811#	813	813#
815	815#	817	817#	819	819#	822	822#	824	824#	826	826#	828	828#
830	830#	832	832#	834	834#	836	836#	838	838#	840	840#	842	842#
844	844#	846	846#	848	848#	850	850#	852	852#	857	857#	859	859#
862	862#	864	864#	866	866#	868	868#	870	870#	872	872#	874	874#
876	876#	878	878#	880	880#	882	882#	884	884#	886	886#	888	888#
890	890#	892	892#	895	895#	897	897#	899	899#	901	901#	903	903#
905	905#	907	907#	909	909#	911	911#	913	913#	915	915#	917	917#
919	919#	921	921#	923	923#	925	925#	928	928#	930	930#	932	932#
934	934#	936	936#	938	938#	940	940#	942	942#	944	944#	946	946#
948	948#	950	950#	952	952#	954	954#	956	956#	958	958#	961	961#
963	963#	965	965#	967	967#	969	969#	971	971#	973	973#	975	975#
977	977#	979	979#	981	981#	983	983#	985	985#	987	987#	989	989#
991	991#	996	996#	998	998#	1001	1001#	1003	1003#	1005	1005#	1007	1007#
1009	1009#	1011	1011#	1013	1013#	1015	1015#	1017	1017#	1019	1019#	1021	1021#
1023	1023#	1025	1025#	1027	1027#	1029	1029#	1031	1031#	1034	1034#	1036	1036#
1038	1038#	1040	1040#	1042	1042#	1044	1044#	1046	1046#	1048	1048#	1050	1050#
1052	1052#	1054	1054#	1056	1056#	1058	1058#	1060	1060#	1062	1062#	1064	1064#
1067	1067#	1069	1069#	1071	1071#	1073	1073#	1075	1075#	1077	1077#	1079	1079#
1081	1081#	1083	1083#	1085	1085#	1087	1087#	1089	1089#	1091	1091#	1093	1093#
1095	1095#	1097	1097#	1100	1100#	1102	1102#	1104	1104#	1106	1106#	1108	1108#
1110	1110#	1112	1112#	1114	1114#	1116	1116#	1118	1118#	1120	1120#	1122	1122#
1124	1124#	1126	1126#	1128	1128#	1130	1130#	1135	1135#	1137	1137#	1140	1140#
1142	1142#	1144	1144#	1146	1146#	1148	1148#	1150	1150#	1152	1152#	1154	1154#
1156	1156#	1158	1158#	1160	1160#	1162	1162#	1164	1164#	1166	1166#	1168	1168#
1170	1170#	1173	1173#	1175	1175#	1177	1177#	1179	1179#	1181	1181#	1183	1183#
1185	1185#	1187	1187#	1189	1189#	1191	1191#	1193	1193#	1195	1195#	1197	1197#
1199	1199#	1201	1201#	1203	1203#	1206	1206#	1208	1208#	1210	1210#	1212	1212#

SEQ 1247

1214 1214# 1216 1216# 1218 1218# 1220 1220# 1222 1222# 1224 1224# 1226 1226#

SEQ 1235

1214	1214#	1216	1216#	1218	1218#	1220	1220#	1222	1222#	1224	1224#	1226	1226#
1228	1228#	1230	1230#	1232	1232#	1234	1234#	1236	1236#	1239	1239#	1241	1241#
1243	1243#	1245	1245#	1247	1247#	1249	1249#	1251	1251#	1253	1253#	1255	1255#
1257	1257#	1259	1259#	1261	1261#	1263	1263#	1265	1265#	1267	1267#	1269	1269#
1274	1274#	1276	1276#	1279	1279#	1281	1281#	1283	1283#	1285	1285#	1287	1287#
1289	1289#	1291	1291#	1293	1293#	1295	1295#	1297	1297#	1299	1299#	1301	1301#
1303	1303#	1305	1305#	1307	1307#	1309	1309#	1312	1312#	1314	1314#	1316	1316#
1318	1318#	1320	1320#	1322	1322#	1324	1324#	1326	1326#	1328	1328#	1330	1330#
1332	1332#	1334	1334#	1336	1336#	1338	1338#	1340	1340#	1342	1342#	1345	1345#
1347	1347#	1349	1349#	1351	1351#	1353	1353#	1355	1355#	1357	1357#	1359	1359#
1361	1361#	1363	1363#	1365	1365#	1367	1367#	1369	1369#	1371	1371#	1373	1373#
1375	1375#	1378	1378#	1380	1380#	1382	1382#	1384	1384#	1386	1386#	1388	1388#
1390	1390#	1392	1392#	1394	1394#	1396	1396#	1398	1398#	1400	1400#	1402	1402#
1404	1404#	1406	1406#	1408	1408#	1413	1413#	1415	1415#	1418	1418#	1420	1420#
1422	1422#	1424	1424#	1426	1426#	1428	1428#	1430	1430#	1432	1432#	1434	1434#
1436	1436#	1438	1438#	1440	1440#	1442	1442#	1444	1444#	1446	1446#	1448	1448#
1451	1451#	1453	1453#	1455	1455#	1457	1457#	1459	1459#	1461	1461#	1463	1463#
1465	1465#	1467	1467#	1469	1469#	1471	1471#	1473	1473#	1475	1475#	1477	1477#
1479	1479#	1481	1481#	1484	1484#	1486	1486#	1488	1488#	1490	1490#	1492	1492#
1494	1494#	1496	1496#	1498	1498#	1500	1500#	1502	1502#	1504	1504#	1506	1506#
1508	1508#	1510	1510#	1512	1512#	1514	1514#	1517	1517#	1519	1519#	1521	1521#
1523	1523#	1525	1525#	1527	1527#	1529	1529#	1531	1531#	1533	1533#	1535	1535#
1537	1537#	1539	1539#	1541	1541#	1543	1543#	1545	1545#	1547	1547#	1552	1552#
1554	1554#	1557	1557#	1559	1559#	1561	1561#	1563	1563#	1565	1565#	1567	1567#
1569	1569#	1571	1571#	1573	1573#	1575	1575#	1577	1577#	1579	1579#	1581	1581#
1583	1583#	1585	1585#	1587	1587#	1590	1590#	1592	1592#	1594	1594#	1596	1596#
1598	1598#	1600	1600#	1602	1602#	1604	1604#	1606	1606#	1608	1608#	1610	1610#
1612	1612#	1614	1614#	1616	1616#	1618	1618#	1620	1620#	1623	1623#	1625	1625#
1627	1627#	1629	1629#	1631	1631#	1633	1633#	1635	1635#	1637	1637#	1639	1639#
1641	1641#	1643	1643#	1645	1645#	1647	1647#	1649	1649#	1651	1651#	1653	1653#
1656	1656#	1658	1658#	1660	1660#	1662	1662#	1664	1664#	1666	1666#	1668	1668#
1670	1670#	1672	1672#	1674	1674#	1676	1676#	1678	1678#	1680	1680#	1682	1682#
1684	1684#	1686	1686#	1691	1691#	1693	1693#	1696	1696#	1698	1698#	1700	1700#
1702	1702#	1704	1704#	1706	1706#	1708	1708#	1710	1710#	1712	1712#	1714	1714#
1716	1716#	1718	1718#	1720	1720#	1722	1722#	1724	1724#	1726	1726#	1729	1729#
1731	1731#	1733	1733#	1735	1735#	1737	1737#	1739	1739#	1741	1741#	1743	1743#
1745	1745#	1747	1747#	1749	1749#	1751	1751#	1753	1753#	1755	1755#	1757	1757#
1759	1759#	1762	1762#	1764	1764#	1766	1766#	1768	1768#	1770	1770#	1772	1772#
1774	1774#	1776	1776#	1778	1778#	1780	1780#	1782	1782#	1784	1784#	1786	1786#
1788	1788#	1790	1790#	1792	1792#	1795	1795#	1797	1797#	1799	1799#	1801	1801#
1803	1803#	1805	1805#	1807	1807#	1809	1809#	1811	1811#	1813	1813#	1815	1815#
1817	1817#	1819	1819#	1821	[illegible]	1823	1823#	1825	1825#	1830	1830#	1832	1832#
1835	1835#	1837	1837#	1839	1839#	1841	1841#	1843	1843#	1845	1845#	1847	1847#
1849	1849#	1851	1851#	1853	1853#	1855	1855#	1857	1857#	1859	1859#	1861	1861#
1863	1863#	1865	1865#	1868	1868#	1870	1870#	1872	1872#	1874	1874#	1876	1876#
1878	1878#	1880	1880#	1882	1882#	1884	1884#	1886	1886#	1888	1888#	1890	1890#
1892	1892#	1894	1894#	1896	1896#	1898	1898#	1901	1901#	1903	1903#	1905	1905#
1907	1907#	1909	1909#	1911	1911#	1913	1913#	1915	1915#	1917	1917#	1919	1919#
1921	1921#	1923	1923#	1925	1925#	1927	1927#	1929	1929#	1931	1931#	1934	1934#
1936	1936#	1938	1938#	1940	1940#	1942	1942#	1944	1944#	1946	1946#	1948	1948#
1950	1950#	1952	1952#	1954	1954#	1956	1956#	1958	1958#	1960	1960#	1962	1962#
1964	1964#	1969	1969#	1971	1971#	1974	1974#	1976	1976#	1978	1978#	1980	1980#
1982	1982#	1984	1984#	1986	1986#	1988	1988#	1990	1990#	1992	1992#	1994	1994#

SEQ 1248